A Machine Learning Approach to Anaphora Resolution Including Named Entity Recognition, PP Attachment Disambiguation, and Animacy Detection

Anders Nøklestad

May 7, 2009

Series of dissertations submitted to the
Faculty of Humanities, University of Oslo
No. 397

ISSN 0806-3222

Cover: Inger Sandved Anfinsen.
Printed in Norway: AiT e-dit AS, Oslo, 2009.

Produced in co-operation with Unipub AS.
The thesis is produced by Unipub AS merely in connection with the
thesis defence. Kindly direct all inquiries regarding the thesis to the copyright
holder or the unit which grants the doctorate.

Unipub AS is owned by
The University Foundation for Student Life (SiO)

For my parents, Randi and Hans Olaf

Contents

Preface

My desire to write this thesis emerged from some of the needs that have surfaced in the course of my time as a programmer and systems engineer at the Text Laboratory located at the University of Oslo. At the Text Lab, we work on creating language resources and making them available to the community of linguistic researchers, with a main focus on resources for the Norwegian language. Being a relatively small language, Norwegian has not been blessed with the amount of tools and resources that exist for more widespread languages. Hence, I saw the need for many more Norwegian resources, and I thought that someone should take the time to develop some of those resources—and that person could be me.

In doing this work, I have been able to combine the creation of much-needed Norwegian language resources with an exploration into a number of very exciting statistical methods for doing machine learning of language. This work has turned out to show new and surprising results for some of the tested methods. The result is a combination of a monograph and a set of—hopefully valuable—tools and resources for Norwegian.

Acknowledgements

First and foremost, I want to thank my primary supervisor, professor Janne Bondi Johannessen, and my secondary supervisor, professor Christer Johansson, for their continuous inspiration and guidance that made it possible for me to finally complete this thesis. Janne's keen eye, sharp mind, and extensive knowledge about language and linguistics were invaluable in making this thesis into what it has become.

Christer's expertise in anaphora resolution and data-driven NLP methods gave me the necessary foundation to go through with this project. Our many discussions about these topics have contributed immensely to my work, and the development of the machine learning-based anaphora resolution in particular has benefited from Christer's ideas and suggestions.

I also want to extend my heartfelt thanks to my colleagues at the Text Laboratory, in particular Janne (again), Kristin Hagen, Joel Priestley, Lars Nygaard, and Åshild Søfteland, for making the Text Lab such an inspirational, pleasant, and fun place to work in.

My *hovedfag* supervisor, Hanne Gram Simonsen, was instrumental in making me want to pursue an academic career. Without her encouragement, I probably never would have ventured into doing a PhD.

I want to thank the *Induction of Linguistic Knowledge* group at Tilburg University and their colleagues at the University of Antwerp for accepting me as a visiting student in Tilburg and for providing me with a very inspirational environment for a few months during my PhD work. In particular, I want to thank Antal van den Bosch for sharing his extensive knowledge about data-driven methods and for making me feel very welcome in Tilburg, and I thank Walter Daelemans and Veronique Hoste for the valuable discussions we had about anaphora resolution and other aspects of my work. Also, a big thanks to the other members of the group who made my stay in Tilburg a pleasant one, in particular Martin Reynaert, Iris Hendrickx, Ielka van der Sluis, and Piroska Lendvai.

I thank Kaja Borthen and Lars G. Johnsen for annotating the anaphora resolution data, Lars Nygaard for co-annotating the PP attachment data with me, and Åshild Søfteland for evaluating my lists of animate nouns. I also thank Gordana Ilić Holen for

providing me with information about her anaphora resolution work and for discussions about the specifics of our respective AR systems. Chris Biemann graciously provided frequency statistics for syntactic relations in the Oslo Corpus.

And last, but definitely not least, I want to thank Kristin Norstad for being my closest friend and soulmate for so many years. Thank you for believing in me all this time, and for being the best friend a man could have. I never would have got to this point without you, Kristin!

The initial phases of the named entity recognition work described in this thesis were carried out within the Nomen Nescio network. Nomen Nescio was a Scandinavian named entity research network which lasted from 2001 to 2003 and was funded by the Nordic Council of Ministers through *Nordisk forskerutdanningsakademi* (NorFA). The network was organized by Prof. Janne Bondi Johannessen. I thank the Faculty of Humanities at the University of Oslo for providing me with the funding to do the rest of my PhD work.

My anaphora resolution work was carried out in close co-operation with the BREDT (Processing Reference from Discourse Theory) project, a project for coreference resolution in Norwegian, organized by Prof. Christer Johansson. The BREDT project is a project within the KUNSTI (Knowledge Development for Norwegian Language Technology) programme, supported by the Research Council of Norway (NFR).

Chapter 1

Introduction

1.1 The main aim: Automatic anaphora resolution

The main aim of this thesis is the development of an automatic anaphora resolution (AR) system for Norwegian. Anaphora resolution (and the related, more general task of coreference resolution) is about determining what words or phrases in a text refer to the same entity. Anaphora resolution deals with pairs of such *coreferent* phrases. One of these phrases, the *anaphor*, consists of a pronoun or an adverb. The other one, the *antecedent*, provides the information required to interpret the anaphor. Most often, the antecedent takes the form of a noun phrase (NP), but it might also be a clause, an entire sentence, or even a sequence of sentences. In the present thesis, as in most work on anaphora resolution, only NP antecedents are taken into account.

The following example illustrates the problem of anaphora resolution:

(1) Toget traff reinsdyret fordi ...
train-the hit reindeer-the because
"The train hit the reindeer because ..."

a. det kjørte for fort.
it drove too fast
"it was driving too fast."

b. det sto i skinnegangen.
it stood in rails-the
"it was standing on the rails."

c. det var mørkt.
it was dark
"it was dark."

In all of the clauses a–c, the first word, *det* "it" is a potential anaphor because it is a personal pronoun (third-person neuter pronoun). However, only in the first two clauses

does it have an antecedent: in (1-a), *det* "it" corefers with (has as its antecedent) *Toget* "the train", while in (1-b), it corefers with *reinsdyret* "the reindeer". In (1-c), on the other hand, *det* does not corefer with anything. The task of an automatic anaphora resolution system is to determine whether a potentially anaphoric expression is in fact an anaphor in any given case, and, if so, to determine its antecedent.

Note that, in many cases, an anaphor will be part of a coreference chain, meaning that it corefers with a whole set of expressions found earlier in the text. Although each of these expressions may be said to play the role of antecedent to the anaphor in question, when I refer to *the* antecedent of an anaphor in this thesis, I am always referring to its *closest* antecedent. This definition of antecedent is suitable in the context of automatic anaphora resolution, since the closest antecedent is normally what an automatic AR system is required to find.

1.2 Support modules

In order for a computer program to do anaphora resolution, it requires access to various types of information about the text it is working on. Some of these types of information are morphological information, syntactic information, and a certain amount of semantic information—most notably, information about whether potential antecedents are animate or inanimate.

For Norwegian, tools for analyzing texts and establishing some of the required information already exist. This is the case, for instance, for morphological and syntactic information, which is provided by the Oslo-Bergen grammatical tagger (cf. section 3.3).

There are, however, various other types of potentially useful information for which no existing tools are available. For this reason, considerable parts of this thesis deal with descriptions of tools that I have developed in order to obtain some of that information: a named entity recognizer (chapter 4), which classifies proper names into various categories, a prepositional phrase (PP) attachment disambiguator (chapter 5), which determines whether a prepositional phrase modifies the verb or the OBJECT of a sentence, and techniques for extracting information about the animacy of Norwegian nouns from the World Wide Web (chapter 6).

In addition to serving as support modules for the AR system, the named entity recognizer, the PP attachment disambiguator, and the lists of of animate nouns obtained from the Web fulfil important functions on their own, and can be used as standalone tools and resources in other kinds of natural language processing (NLP) applications. In the course of their development, I have also investigated the use of various machine learning techniques as well as the effect of feature selection and parameter optimization. For these reasons, considerable space has been allocated to the description of these tools.

1.2.1 Animacy detection

Since some pronouns, such as *han* "he" and *hun* "she" typically refer to humans, while others, such as *den* "it (masc./fem.)" and *det* "it (neut.)", normally refer to non-humans, knowing which of the expressions in a text refer to humans may be very important in order to determine suitable antecedent candidates for an anaphor.

Note that, since animals (pets in particular) as well as humans may be referred to by pronouns such as *han* and *hun*, I use the term *animacy* rather than something like "humanness" in this thesis. Nevertheless, my techniques for detecting animacy are clearly geared towards discovering expressions that refer to humans, since the expressions that corefer with pronouns like *han* and *hun* are most often of this kind.

The following example illustrates the usefulness of animacy detection:

(2) Den eldre mannen satte seg inn i bilen. Etter noen forsøk
the elderly man sat himself in in car-the after some attempts
klarte han å få startet den.
managed he to get startet it
"The elderly man got into the car. After a few attempts, he managed to start it."

In order to determine the antecedents of *han* "he" and *den* "it" in the second sentence, it is very useful to know that *mann* "man" is animate and hence a suitable antecedent candidate for *han* but not for *den*, while for the inanimate noun *car* it is the other way round.

1.2.2 Named entity recognition

The field of named entity recognition has received a fair amount of attention over the last decade. This attention was to a large extent triggered by the last two Message Understanding conferences and competitions, MUC-6 (Grishman and Sundheim, 1996) and MUC-7 (Chinchor, 1997), which included named entity recognition as a separate task.

Named entity recognition (NER) is the task of identifying named entities in a text and classifying them into a (usually) predefined set of categories. What constitutes a named entity varies among different projects, but proper names are always included. Some definitions of the task also include date and time expressions as well as various numerical expressions among the named entities. This was the case, for instance, in the MUC competitions and in the IREX project (Sekine and Isahara, 2000), but not in the ACE project (ACE, 2000) or in the Nomen Nescio project (Johannessen, Hagen, Haaland, Jónsdottir, Nøklestad, Kokkinakis, Meurer, Bick, and Haltrup, 2005), which constitutes the context of the NER work described in this thesis.

Knowing whether an expression denotes a named entity, in particular a proper name, and, if so, what kind of named entity it denotes, is very helpful for the task of anaphora resolution. Consider the following example:

(3) Carl lå i senga og leste Idioten. Han likte den godt.
Carl lay in bed-the and read Idiot-the he liked it well
"Carl lay in bed reading the Idiot. He enjoyed it."

In this case, it is not sufficient to determine the animacy of the various nouns in the sentence. Animacy detection may help us determine that *senga* "the bed" is a suitable (albeit incorrect) antecedent candidate for *den*, but not for *Han*. It cannot tell us, however, that, while *Carl* is indeed a suitable antecedent for *Han*, *Idioten* is not, because an idiot is an animate entity. Using named entity recognition, on the other hand, we should be able to determine that *Idioten* is in fact a work of art and hence not an animate entity in this context, leaving us with *Carl* as the only appropriate antecedent for *Han*.

1.2.3 PP attachment disambiguation

PP attachment disambiguation is the task of determining, for a certain prepositional phrase, what other word or constituent in the sentence it attaches to, or modifies. Typically, as in the present thesis, it is stated as the task of deciding whether a PP attaches to the main verb of the clause or to a noun that occurs between the verb and the PP itself. In some of the previous work on anaphora resolution, it has been argued that a noun which is found in a PP that attaches to another noun, and which therefore becomes embedded within the NP headed by the second noun, is a less likely antecedent candidate than nouns that are not embedded in this way. Consider the following example:

(4) a. Ivan så på korken på flaska. Den var blå.
Ivan saw on cork-the on bottle-the it was blue
"Ivan looked at the cork on the bottle. It was blue."

b. Ivan satte korken på flaska. Den var blå.
Ivan put cork-the on bottle-the it was blue
"Ivan put the cork on the bottle. It was blue."

In (4-a), the PP *på flaska*, and hence the noun *flaska*, is embedded in the OBJECT NP *korken på flaska* "the cork on the bottle". In (4-b), on the other hand, the PP is an ADVERBIAL and hence not embedded. While *korken/Korken* might be the most likely antecedent for *Den* in both cases (which may be due to the salience of OBJECTs

compared to prepositional complements), the fact that *flaska* is embedded in (4-a) but not in (4-b) might nevertheless make it a more likely antecedent in the latter sentence pair than in the former, at least if the embeddedness principles employed by, for example Lappin and Leass (1994) hold true. In the present thesis, I investigate whether PP attachments determined by an automatic PP attachment disambiguator can contribute to the anaphora resolution task on Norwegian fiction.

1.3 Research questions

This thesis sets out to investigate a number of research questions. The questions can be summed up as follows:

- How well do the most commonly used knowledge-poor approaches to anaphora resolution—machine learning, factor/indicator-based approaches, and Centering Theory—work for Norwegian?
- How well do each of these approaches work on fiction material?
- How well do different machine learning approaches perform on this task?
- How can we use machine learning methods to produce some linguistic information that is potentially valuable for anaphora resolution, more specifically information about
  - named entity types
  - PP attachment
  - animacy
- Does the inclusion of named entity recognition, PP attachment disambiguation, and animacy detection contribute to better results for anaphora resolution?
- Can we find improved methods for obtaining training data for supervised machine learning methods?
- Can we improve on the performance of a machine learning-based anaphora resolution system by utilizing techniques inspired by other knowledge-poor approaches?

1.4 Main contributions

Due to its treatment of various supporting technologies as well as the anaphora resolution task itself, this thesis contributes in a variety of ways to the field of natural language processing. The main contributions of the thesis can be summed up in the following way.

For named entity recognition (NER), I compare the use of a "lazy" method, memory-based learning (MBL), and an "eager" method, maximum entropy (MaxEnt) modelling. I divide the NER task into an identification stage and a classification stage, and apply the machine learning method only to the classification stage (using a set of rules developed by the Nomen Nescio NER project for the identification part). Using this setup, I find that, contrary to earlier reports in the literature, the memory-based learner performs very well on the NER task, actually outperforming the MaxEnt model under some conditions. I develop a kind of document-centred post-processing that takes classifier confidence into account, and show that it helps improve performance even further. I also evaluate the effect of different machine learning features on this task, and investigate the impact of manual and automatic parameter optimization.

In evaluating the performance of the memory-based learner on the NER task, I find that the leave-one-out testing procedure reports a much better performance level than 10-fold cross validation. However, I point out certain problems with leave-one-out testing when it comes to tasks such as NER, and argue that this testing methodology is not really appropriate for the NER task or for other tasks in which correct classification depends on properties of the document on which classification takes place (in the NER case, classification is dependent on the classification of other names within the particular document that the test item occurs in).

For the PP attachment disambiguation task, I introduce a new way to seed a supervised learner by automatically extracting examples from a corpus which has not been manually annotated for PP attachment. I compare the use of memory-based learning, maximum entropy modelling, and support vector machines, and investigate the effect of automatic parameter optimization. I also discuss some reasons to doubt the value of the so-called "human-level performance" measure that has long been taken at face value in this field.

In my work on animacy detection, I explore various ways to use queries on the World Wide Web in order to extract lists of animate nouns as well as to check the animacy value of a particular noun. Unlike the earlier research that inspired my work in this area, my own approach produces not only information that is useful for other NLP tasks, but also lists of animate nouns that can be of use even outside the NLP field. My experiments suggest that recall is more important than precision when it comes to gathering animate nouns for anaphora resolution purposes, a promising result for the kind of text-mining approaches described in chapter 6.

My anaphora resolution (AR) system is the first AR system for Norwegian that employs machine learning. To my knowledge, it is also the first AR system to focus exclusively on fiction texts (most earlier systems get their material from selected newspapers or magazines or from technical writing, although some systems, such as Tetreault (2001) and Holen (2006), have also been tested on fiction material in addition to other types of text).

Once again, I compare the use of different machine learning methods on the task. Additionally, I compare the performance of the machine learning methods to that of two alternative approaches: a Centering Theory approach and a factor-based approach. To my knowledge, this is the first systematic comparison of these knowledge-poor approaches on the same data, preprocessed in exactly the same way and using exactly the same set of anaphors and antecedent candidates. Identical conditions for each method are guaranteed by the fact that the algorithms are implemented as software modules which are plugged into the same overall AR architecture.

For the machine learning approach to anaphora resolution, I demonstrate the usefulness of employing different classifiers for different types of pronouns, thereby going a step further than previous AR research in separating the problem domain into different tasks handled by different classifiers. I also test the value of the named entity, PP attachment, and animacy information that is obtained using the tools described in the earlier chapters, and investigate a combination of machine learning classification with an antecedent search strategy that is inspired by Centering-based approaches.

Finally, using fiction material rather than news texts or technical writing has unveiled some differences with respect to antecedent distribution in these text types. Specifically, in fiction, a character is often introduced by a pronoun that has no antecedent in the text. Similarly, plural pronouns are often used generically. Also, animate pronouns are regularly separated from their antecedents by one or more alternative, but false, candidates. These facts are likely to have a negative impact on the performance of the anaphora resolution system.

1.5 Some areas of application

All of the tools developed in this thesis can contribute to the improvement of systems that either rely completely on natural language processing or that could at least benefit from such processing. This includes systems like corpus query tools that are used by linguists who study the Norwegian language. It also pertains to systems that are used by the general Norwegian-speaking public when they communicate with computer software and with various services such as phone-based systems for information enquiries, ticket bookings, and so on.

In the following sections, I take a brief look at some of the fields where language

technology can contribute to making existing systems easier to use, and discuss what kind of contributions we can expect from anaphora resolution systems such as the one described in chapter 8, as well as directly from the support modules that are presented in earlier chapters.

1.5.1 Information retrieval

The World Wide Web has grown into a huge repository of information, and for many people it has become the first place they look for information on virtually any subject. With such large amounts of information available, the biggest problem becomes that of finding those pieces of information that are (most) relevant for some particular query. Getting hold of that information is the subject of the fields of *information retrieval*, *information extraction*, and *question answering*.

Information retrieval (IR) is the most comprehensive of these terms. It refers to the general task of retrieving some kind of information from an information store, e.g., an online collection of texts, a database of patient records, or a physical library (which was the domain of the original IR systems (Sparck Jones and Willett, 1997)). In recent years, however, it has been mostly used to refer to the task of retrieving electronic documents from a document collection, usually the World Wide Web.

Unlike the data in many databases and physical libraries, documents on the Web are not pre-classified and tagged with meta-data that can be used for retrieval. Instead, IR systems for the Web have to employ methods that, given one or more search terms, search through the collection and find those documents that are most relevant in the light of the given search terms, where the degree of relevance is determined by some particular calculation. This is the kind of method that is employed by popular search engines on the Web, such as Google and Alta Vista.

Information retrieval on the Web is most valuable for users that are interested in some particular subject matter and want to find a set of documents that provide further information about that subject. However, if we are instead interested in extracting specific pieces of information from the documents, perhaps in order to register this information in a database, to feed it into some system that performs a higher-level language processing task, or simply to get a concise answer to a particular question, then we need to explore the fields of *information extraction* and, in the latter case, *question answering*.

1.5.2 Information extraction

Compared to information retrieval, information extraction (IE) goes a step further and aims at extracting only the relevant information from a set of documents, rather than retrieving the actual documents in their entirety. In fact, several of the tasks

described in this thesis—named entity recognition, PP attachment disambiguation, and anaphora resolution—can be seen as examples of IE tasks.

In named entity recognition, the goal is to extract certain kinds of entities from the texts, most notably proper names, but sometimes also dates and numerical expressions. In PP attachment disambiguation, we extract information about which word or constituent in a sentence a certain prepositional phrase attaches to. In anaphora resolution, we want to obtain information about which constituents act as the antecedents of a certain anaphoric expression. Moving one level upwards, all of these IE subtasks are crucial components in higher-level systems such as question answering systems and machine translation systems.

1.5.3 Question answering

Search engines like Google and Lycos, which perform information retrieval on the Web, are extremely valuable for anybody looking for documents about a certain subject matter. Very often, however, what we really want is to get concise answers to certain questions. In those cases, we do not really want to have to read through a number of documents which might be relevant to our question, but that nevertheless contain a large amount of additional information besides the answers that we are looking for. In those cases, what we really want is a question answering (QA) system, i.e., a system that accepts our question (preferably formulated in a natural language like English or Norwegian) and returns a concise answer (or set of answers) to that question.

Although this is probably the kind of system that would be most valuable to most people most of the time, it is at the same time a very difficult kind of system to build. Such a system has to be able to analyze and, to a certain extent, "understand" the question posed by the user. It then has to retrieve the set of documents that are relevant to the user's question and identify the answer(s) to the question in those documents, and return those answers to the user. As hard as this task may be, the obvious benefits of such a system make it a highly attractive goal to reach towards. For QA systems to obtain performance levels that are high enough to make them really useful in practice, however, a large number of tasks need to be solved, including those that are dealt with in this thesis.

1.5.4 Machine translation

Among the fields that have attracted the most interest from NLP researchers over the past fifty years is the field of machine translation. Translation is a time-consuming and demanding task, and there are many areas of society that pose a continuous need for texts to be translated into several languages. Some of these are hardly eligible for automatic translation, such as fiction material, which will probably always require the

skills of a human translator. However, many other types of text can be (and sometimes are) translated by machine translation systems.

One example is appliance manuals, which need to be translated into the languages of the different countries in which the appliance is to be sold. Another example is official texts created by the European Union (declarations, resolutions, etc.), which have to be translated into the different languages spoken by the members of the union.

Machine translation is a very difficult task, however, as is evident from the fact that, after fifty years of research, the translations produced by even the best systems are still a far cry from translations produced by a skilled human translator. This has to do with the fact that, in order to produce a good translation, a system has to master most aspects of both source and target language, including having the required morphological, syntactic, semantic, and pragmatic knowledge. The mechanisms developed in this thesis can provide some small but important parts of the knowledge required by machine translation systems.

1.5.5 How the present work can be useful

For information extraction and question answering purposes, the link that is established between an anaphor and its antecedent (as well as further expressions that corefer with the antecedent) may provide additional information that can be extracted. Imagine that we have a question answering system at our disposal and ask it the following question: *When was John Lennon killed?*. Imagine further that the following sequence is found in one of the documents available to the system:

(5) John Lennon was a great musician. He was killed in 1980.

As human readers, we immediately see that this sequence answers our question: John Lennon was killed in 1980. However, this deduction actually requires that a link be established between *John Lennon* in the first sentence and *He* in the second, so that the fact about being killed in 1980 that is mentioned in the second sentence can be attributed to *John Lennon* in the first.

Anaphora resolution is also valuable for machine translation, because the usage patterns of different pronouns vary between languages. For instance, English and Norwegian Bokmål use one type of pronoun, *he/han* and *she/hun*, to refer to humans (and personified animals, mythical figures, etc.) and another, *it/den/det*, to refer to other entities. In Norwegian Nynorsk, on the other hand, the pronoun used to refer to masculine humans is used to refer to all other masculine nouns as well, and vice versa for females, while the distinctly non-human pronoun *det* is restricted to neuter nouns. Thus, in order to translate correctly between English or Norwegian Bokmål on the one hand and Norwegian Nynorsk on the other, we cannot do a simple one-to-one

translation of pronouns. Rather, we need to establish the antecedent of each pronoun to see how it should be translated in each individual case.

Identifying and classifying named entities is required for information retrieval, information extraction, and question answering, particularly in cases where a named entity occurs as one of the search terms or as the topic of the question asked. To begin with, if the search word could be either a common name or a proper name, we would like to make sure that we search for the right type of noun, which means that we need to be able to identify those cases where the word is a proper name so that we can either include or exclude it from the information returned by the system.

For example, imagine that we want to search Google or another search engine for documents with information about how to become a baker. Considering the widespread occurrence of the surname *Baker* and the fact that popular search engines perform case-insensitive searches, we would probably have to wade through a large amount of documents in order to identify those that are actually about the baking profession.

Of course, with current search engines we could try to filter out irrelevant documents by requiring that documents contain additional search terms having to do with becoming a baker, such as *school* and *education*. However, there might be relevant documents that do not contain any of these additional terms and hence will be excluded from the results. A better approach would be to identify proper-name occurrences of *Baker* and exclude them from the information considered relevant to the query. Conversely, if we want to search for information about, say, the actress Minnie Driver, it would be good if the system could identify only those documents where *Driver* occurs as a proper name and exclude documents that mention drivers in general.

Thus, we see that the NER subtask of named entity *identification* has potential benefits for IR, IE, and QA systems. However, the second NER subtask, named entity *classification*, is equally important for cases where a proper name might have several different kinds of referents. For example, the name *Java* might refer to an island, a programming language, or a coffee brand, and a good search engine should give us the opportunity to choose which of these we are interested in. For example, considering the abundance of computer-related documents on the Web and the widespread use of the Java programming language, submitting the single query term *Java* to a search engine is likely to return a vast majority of documents about the programming language, making it very difficult to spot those presumably few documents that are about the coffee brand.

As another example, consider the name *Ericsson*, which might refer to a person or a company. The executives of the company might be interested in finding documents on the Web, particularly newspaper articles, that mention the company, and it would be of great value to them to have documents about *people* called Ericsson filtered out.

When it comes to machine translation, named entity identification is probably more important than named entity classification. As is the case with the IR/IE/QA tasks, the relevance of NE identification for machine translation becomes apparent when a text contains words (or phrases) that can be either proper names or common nouns. For example, the Norwegian noun *mann* should normally be translated to *man* in English, but only if it is used as a common name, as in (6-a), and not as, for instance, the name of the author Thomas Mann, as in (6-b).

(6) a. Det sitter en gammel mann på trappa.
it sits an old man on stairs-the
"An old man is sitting on the stairs."

b. Han har ikke lest noenting av Thomas Mann.
he has not read anything by Thomas Mann
"He has not read anything by Thomas Mann."

The benefits of PP attachment disambiguation for higher-level tasks might be less directly obvious than those of named entity recognition. However, the attachment site of a PP determines which constituent the PP provides information about, and hence it might be important for information extraction and question answering tasks.

If a PP attaches to a preceding noun, the information conveyed by the PP is part of the total information that is available for that noun, and hence it should be taken into consideration when we need to extract information about the noun. If, on the other hand, the PP attaches to the main verb of the sentence, the information it provides pertains to the event denoted by the verb instead[1].

Furthermore, PP attachment disambiguation is important for systems that require a certain level of language understanding. Syntactic parsers will often have problems disambiguating PP attachment, creating syntactic parse trees where PP attachment is left ambiguous or dependency structures which do not specify the head which the PP depends on. Thus, in order to obtain complete syntactic parse trees or dependency structures, we need to solve this task.

Finally, PP attachment disambiguation is required for natural-sounding speech generation in speech-based human-computer interfaces, more specifically for determining phrase boundaries that should be expressed as pauses in the generated speech (Marsi, Coppen, Gussenhoven, and Rietveld, 1997; van Herwijnen, Terken, van den Bosch, and Marsi, 2003). For example, in (7-a), where the PP *of the mountain* attaches to the noun *top*, there is no phrase boundary between this noun and the PP, and hence there should be no pause in the generated speech. In (7-b), on the other hand, the PP *for three hours* functions as an adverbial that attaches to the verb *watch*; in this

[1] However, as we will see in chapter 5, the distinction between noun and verb attachment is often not clear.

case there is a prosodic boundary between the noun and the PP, and a pause would normally be deemed appropriate by native speakers (Marsi et al., 1997).

(7) a. Anna saw the top of the mountain.
b. Anna watched TV for three hours.

Thus, proper placement of pauses influences the naturalness of the generated speech, and the degree of naturalness is certain to influence the perceived quality of the service that employs the speech-based interface. Furthermore, correct positioning of pauses will help disambiguate potentially ambiguous PP attachments, thereby clarifying the intended meaning of the generated speech. Conversely, wrongly placed pauses could obfuscate the meaning and lead to confused users. Hence, PP attachment disambiguation plays an important role in speech generation, both for its use in human-computer interaction and otherwise.

Chapter 2

Applied Data-driven Methods

2.1 Overview

A wide range of data-driven methods have been applied to NLP tasks in the past. For my work, I make use of three established methods that have been among the most popular in recent years, yielding state-of-the-art systems for a variety of NLP tasks. Those methods are memory-based learning, maximum entropy modelling, and support vector machines.

Although these methods all achieve very good results, they have fairly different theoretical foundations, which makes it even more interesting to compare their performances on the same tasks. In the following sections, I will give a brief introduction to each of these methods in turn.

Even though I have used all of these methods to varying extents, my main focus has been on the use of memory-based learning. Hence, that is the method that will receive the most attention in the thesis, and it is therefore the one that I will explain in most detail in this chapter.

2.2 Memory-based learning

Memory-based learning (Daelemans, 1995, 1996; Daelemans and van den Bosch, 2005; Daelemans, van den Bosch, and Zavrel, 1999; Daelemans, Zavrel, van der Sloot, and van den Bosch, 2004) is a machine learning technique that derives from the k-nearest neighbour approach (Aha, Kibler, and Albert, 1991; Cover and Hart, 1967; Devijver and Kittler, 1980). Other names that have been used for this kind of learning algorithm are instance-based, exemplar-based, example-based, case-based, analogical, and locally weighted learning.

Memory-based learning (MBL) has been successfully applied to a wide range of

NLP tasks. The following list enumerates a number of such tasks, with references to some of the work in which memory-based learning has been applied.

- part-of-speech tagging (e.g., Marsi, van den Bosch, and Soudi, 2005; van Halteren, Zavrel, and Daelemans, 2001; Zavrel and Daelemans, 1999)
- grammatical relation finding (e.g., Buchholz, 2002; Buchholz, Veenstra, and Daelemans, 1999)
- shallow parsing (combining memory-based tagging, chunking, PP finding, and grammatical relation finding) (e.g., Canisius and van den Bosch, 2004)
- morphological analysis (e.g., Marsi et al., 2005)
- phoneme-to-grapheme conversion (e.g., Decadt, Duchateau, Daelemans, and Wambacq, 2002)
- understanding user utterances in human-computer spoken dialogue systems (e.g., Lendvai, van den Bosch, Krahmer, and Canisius, 2004; van den Bosch, 2005)
- disfluency detection in spoken language (e.g., Lendvai, van den Bosch, and Krahmer, 2003)
- pitch accent placement (e.g., Marsi, Busser, Daelemans, Hoste, Reynaert, and van den Bosch, 2002)
- semantic role labelling (e.g., van den Bosch, Canisius, Daelemans, Hendrickx, and Sang, 2004)
- word sense disambiguation (e.g., Decadt, Hoste, Daelemans, and van den Bosch, 2004; Hoste, Daelemans, Hendrickx, and van den Bosch, 2002; Mihalcea, 2002; Veenstra, van den Bosch, Buchholz, Daelemans, and Zavrel, 2000)
- named entity recognition (e.g., Hendrickx and van den Bosch, 2003; Meulder and Daelemans, 2003; Tjong Kim Sang, 2002b)
- PP attachment disambiguation (e.g., van Herwijnen et al., 2003; Zavrel, Daelemans, and Veenstra, 1997)
- coreference resolution (e.g., Hoste, 2005; Hoste and van den Bosch, 2007)
- information extraction (e.g., Zavrel and Daelemans, 2003)

The intuition behind a memory-based approach is as follows: given a database of *instances* of some kind that have a predetermined category, a memory-based learner can classify new instances by comparing them to those in the database. It does so

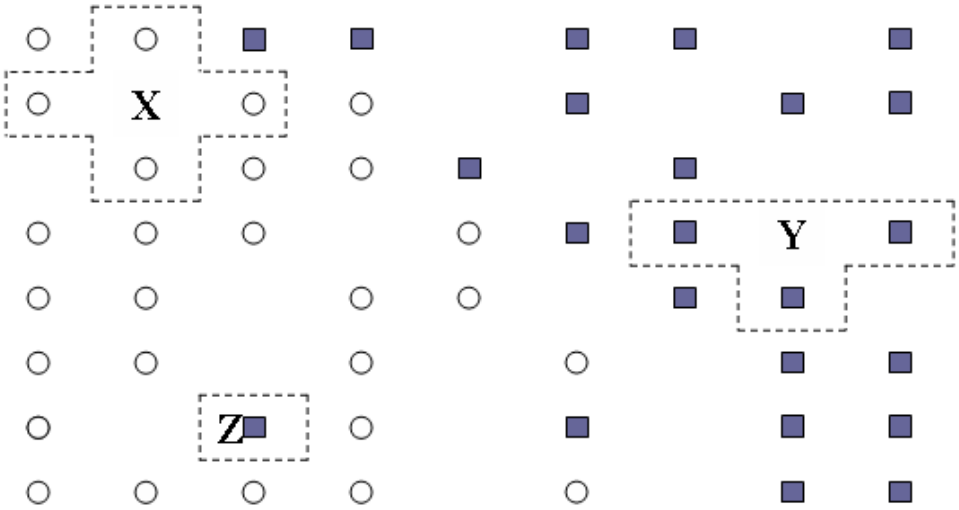

Figure 2.1: Illustration of memory-based classification with $k = 1$. X will be classified as a circle; Y and Z as squares.

by finding those instances in the database that are most similar to the new one and choosing the category that is shared by the majority of them. For NLP tasks, the instances in the database are usually taken from a training corpus, i.e., a corpus that has been manually annotated with the correct category for each instance.

Training a memory-based learner amounts to filling the database with a set of instances along with their annotated category. When a new instance is to be classified, the system will search the database for the instance, or set of instances, that are most similar to the unknown one, i.e., its nearest neighbours (in terms of similarity) in the database. In order to calculate the similarity between instances, we need some *similarity metric*; we will return to the choice of similarity metric in the following sections. The set of nearest neighbours may be called the *support set* for the unknown instance (Daelemans et al., 1999).

The choice of k for the k-nearest classifier determines the size of the support set. If $k = 1$, only those instances that are most similar to the new one are taken into account. If $k = 2$, the next-best matches are also included, and so on. In other words, k refers to the k nearest distances from the unknown instance[1]. The most common choice of k is 1, but under some conditions it is beneficial to choose higher values of k (notably in connection with the use of the Modified Value Difference Metric; see section 2.2.3).

[1]Aha et al. (1991) use the k in k-nearest neighbour to refer to the k nearest *instances* in the database. However, in this work I will interpret k as it is implemented in TiMBL, the software package that I have used in my experiments, i.e., as referring to the k nearest *distances*.

Figure 2.1 gives a simplified illustration of the process of memory-based classification. Here we have visualized the instance database as a vector space in two dimensions. This means that, in this simple example, each instance vector consists of only two elements, which are represented by the horizontal and vertical axis, respectively. All vectors that have the same value for their first element lie along the same vertical axis, while all vectors that have the second element in common lie along the same horizontal axis.

The vector space represents a database of instances that have been preclassified as either circles or squares. Imagine that we want to classify the new instances X, Y, and Z using a k-nearest neighbour approach. Setting $k = 1$, the support set for each of the new instances will be as shown by the stippled areas.

There are no vectors that match X and Y exactly, as we can see from the fact that these instances are located at empty squares in the figure. In other words, neither of these instances has a nearest neighbour in the database which is exactly identical to the instance itself. Rather, their nearest neighbours are vectors that have the same value on one dimension, but not on the other, as shown by the fact that they are found at a distance of one square in either the horizontal or the vertical direction. The support sets of X and Y consist entirely of vectors classified as circles or squares, respectively, with the result that X will be classified as a circle and Y as a square.

Z is located well into a part of the vector space that is dominated by circles. However, because it exactly matches a square, which will then be its sole nearest neighbour, it will also be classified as a square. This case illustrates an important aspect of k-nearest neighbour classifiers that sets them apart from most other machine learning methods: because such classifiers determine the category based on a small neighbourhood of instances, they can be sensitive to "islands" of instances belonging to a certain category within an area of the vector space that is otherwise dominated by instances of another category.

Whether this sensitivity is a good thing largely depends on the characteristics of the phenomenon to be modelled: in many cases, such islands should be considered as noise and should be ignored by the classifier for best performance (this is the approach taken by many of the so-called *eager* methods, such as support vector machines and maximum entropy models). When dealing with language, however, this is often not the case, because all languages contain idiosyncratic phenomena that are not noise, but rather genuine exceptions to the more general patterns or rules that govern the language. We return to this point in section 2.5.1 below.

Note that if we had set k to some value above 1 in Figure 2.1, Z would have been classified as a circle, because its support set would then have contained a number of circles in addition to the square. For example, if we set $k = 2$, the circle to the right of Z and the one below it would both be included in the support set. These two circles

would outnumber the single square, leading to Z being classified as a circle. Thus, the choice of k will strongly influence the classifier's sensitivity to exceptions (or noise).

It should be pointed out that the illustration in Figure 2.1 is a simplified example that leaves out a number of complicating factors. First of all, there will usually be more than a single instance found at each particular location in the vector space. In such cases, the contributions of each instance are added up. Furthermore, the calculation of the support set depends on the choice of *similarity metric*, and the contribution of each instance in the support set is influenced by the kind of *feature weighting* that is used. These factors are discussed in the following sections.

2.2.1 Vector representations

In order to make it possible to compare instances, each instance is represented in terms of a feature vector in which each feature provides some piece of information about the instance that is considered relevant for the current task by the system designer.

To illustrate the kind of vectors that are used in memory-based classification, I will show an example taken from the field of named entity recognition, the topic of chapter 4 of this thesis. An example of a possible (though somewhat simplified) feature vector for a named entity recognizer is given in Table 2.1.

-2	-1	0	1	2	cap	PERSON	LOCATION	ORGANIZATION
be	in	Washington	.	he	yes	yes	yes	no

Table 2.1: A possible feature vector representation of the word *Washington* occurring in the context "Today the president is in Washington. He will leave for New York tomorrow." See the text for details.

The aim of named entity recognition is to predict the categories of proper names (and sometimes other kinds of "named entities"). The vector in Table 2.1 may be used to predict the category of the name *Washington* occurring in the context "Today the president is in Washington. He will leave for New York tomorrow." The name and its context are represented by the following set of features: the lemmas of the tokens that occur one (-1) and two (-2) positions to the left of the name and one (1) and two (2) positions to the right; the lemma of the name itself (0); whether the name is capitalized (cap), and whether the name occurs in gazetteers (i.e., name lists) of people (PERSON), locations (LOCATION), and organizations (ORGANIZATION). See chapter 4 for further details on these features and on named entity recognition in general.

Using vector representation of instances, the similarity between two or more instances is given by the similarity between their vector representations. However, in order to calculate the similarity between two vectors, we need to define an appropriate similarity metric. Commonly used similarity metrics express the degree of similarity as geometric distance in the representational space, which is not necessarily the same

as psychological similarity. The important point is that the similarity metric should yield a high degree of similarity for instances which should be classified in the same way, and a lower similarity for instances that should be classified differently.

One simple metric is the *Overlap metric*. It is given by equations (2.1) and (2.2), where $\Delta(X, Y)$ is the distance between vectors X and Y, n is the number of features in each vector, w_i is the weight of the ith feature, and $\delta(x_i, y_i)$ is the distance between the values of the ith feature in X and Y (Daelemans et al., 1999).

$$\Delta(X, Y) = \sum_{i=1}^{n} w_i \ \delta(x_i, y_i) \tag{2.1}$$

where

$$\delta(x_i, y_i) = 0 \text{ if } x_i = y_i, \text{ else } 1 \tag{2.2}$$

If all features are weighted equally, this metric is simply a count of the number of mismatching features in the two vectors. This is the metric used in the IB1 type of k-nearest neighbour classifier (Aha et al., 1991).

Although assigning equal weight to all features is the simplest method, it is usually not the most appropriate. Consider the vectors in Table 2.2, where we have the same instance as in Table 2.1 at the top and want to classify it by selecting the category of the most similar one of the two stored vectors at the bottom.

-2	-1	0	1	2	cap	PER	LOC	ORG	CAT
be	in	Washington	.	he	yes	yes	yes	no	?
stop	in	Washington	after	having	yes	yes	yes	no	LOC
talk	to	Washington	.	he	yes	yes	yes	no	PER

Table 2.2: At the top, we have an unknown instance that is to be classified by selecting the category of the most similar of the instances below. The categories are found in the CAT column.

The unclassified instance differs from the first stored instance by having values of *be* vs. *stop* for the **-2** feature, full stop vs. *after* for the **1** feature, and *he* vs. *having* for the **2** feature. The second stored instance also differs from the unclassified one with respect to the **-2** feature (*be* vs. *talk*), but here the only other difference is on the **-1** feature (*in* vs. *to*).

Thus, using the Overlap metric with equally weighted features, we find the distance between the first and second vectors to be 3, while the distance between the first and third vectors is 2. Since the third instance is classified as PERSON, the unknown instance will also get a PERSON label, even though this is clearly not correct. Notice that the matches on the full stop and the pronoun following the focus word in the third vector, which are probably not very important for correct classification, count twice

as much as the match on the preposition *in* in the second vector, which presumably is a good indication that the focus word denotes a location. Thus, we also need a way to set different weights on the features according to how important they are for classification.

Note that although in the present example the position immediately to the left of the focus word was the most important one, there may be other contexts in which features pertaining to other positions are more important. For example, in some newspapers, person names are often followed by a number in parentheses giving the person's age, in which case the **1** and **2** features give important disambiguation clues. Commonly used feature-weighting schemes do not weight individual feature values, but rather the feature itself (although they can be used in combination with the Modified Value Difference Metric, which takes into account the similarity between feature values; cf. section 2.2.3). Thus, the weight of a certain feature should reflect its average importance.

2.2.2 Feature weighting

One way of weighting features is to weight the feature by its information gain (Quinlan, 1993). Information gain (IG) is a measure of the reduction in entropy we obtain when we know the value of the feature. The notion of entropy as used in Information Theory (Shannon, 1948) denotes the average number of bits required to transmit the outcome of a random variable across an information channel[2]. The more uncertainty there is about the next outcome of the variable at any given time, the higher the number of bits (i.e., the amount of information) required to transmit it. For example, if we know that the outcome is always the same, no bits at all are required to transmit the outcome. On the other hand, if there are, say, 10 possible outcomes and they are all equally likely, then the average number of bits required for transmission is 3.32.

In the context of MBL, by calculating the information gain of a feature, we compute the difference in uncertainty about the correct classification between a situation where we know the value of this feature and a situation where we do not know it. Information gain is defined as follows (Daelemans et al., 2004):

$$w_i = H(C) - \sum_{v \in V_i} P(v) \times H(C|v) \tag{2.4}$$

[2]The entropy H is defined by the following formula (Shannon, 1948):

$$H = -K \sum_{i=1}^{n} p_i \log p_i \tag{2.3}$$

where K is a positive constant which only determines a unit of measure and may be disregarded for the present purposes, and n is the total number of possible outcomes. In principle, any base can be used for the logarithm, but traditionally base 2 is chosen, and that is what leads to the entropy being expressed as the number of bits (binary pieces of information).

where C is the set of categories, V_i is the set of values for feature i, and $H(C) = -\sum_{c \in C} P(c)\log_2 P(c)$ is the entropy of the categories. The probabilities are estimated from relative frequencies in the training corpus.

There is, however, a problem with IG: if we have a feature with many possible values, the weight of this feature may be overestimated with respect to what is beneficial for generalization—in other words, we might risk overfitting the model to the data. In an extreme case, a feature may have a distinct value for each instance in the database, which will make it very good at reducing the entropy of the category distribution (since it will uniquely identify a single instance). Such a feature will receive a high IG, but it will provide a very poor basis for generalization to unknown instances.

For example, if we assigned an index to each token in the training text and used this index as a feature, knowledge of the index value for a word in the training corpus would completely eliminate our uncertainty about the word's category, but the feature would be useless as a predictor of category membership for instances in new text.

To alleviate this, Quinlan (1993) introduced the notion of *gain ratio*, which normalizes the IG of a feature by its *split info*, the entropy of the distribution of values that the feature can have. If the entropy of a feature value distribution is high, it means that the feature values have similar probabilities of occurring, indicating that they are evenly spread among the training instances and therefore are poor predictors of category membership. Gain ratio (GR) is defined in equations (2.5) and (2.6) (Daelemans et al., 2004).

$$w_i = \frac{H(C) - \sum_{v \in V_i} P(v) \times H(C|v)}{si(i)} \tag{2.5}$$

$$si(i) = -\sum_{v \in V_i} P(v)\log_2 P(v) \tag{2.6}$$

In our token index example, each index value occurs with equal frequency (i.e., 1) in the training corpus, and so this feature would have a very high split info value and hence a low weight.

Although the use of gain ratio reduces the risk of overfitting, White and Liu (1994) have shown that this weighting measure is still biased in favour of features with more values, and they propose a weighting measure based on the chi-squared statistic to correct this. However, as pointed out by Daelemans et al. (2004), this measure does not work very well if the data contain many low-frequent feature values (leading to many low values in the contingency table used in the calculation of the chi-squared statistic). Chi-squared can, however, be improved by correcting for degrees of freedom, yielding a shared variance measure (Daelemans et al., 2004).

In the experiments presented in this thesis, gain ratio weighting turned out to give

the best results, with neither the chi-squared nor the shared variance measures leading to any improvements. Hence, I will not go into further details here about these latter measures.

2.2.3 Modified Value Difference Metric (MVDM)

K-nearest neighbour learning works with features that have either numerical or symbolic values. For numerical features, we can modify equation (2.2) in section 2.2.1 to use the absolute value of the difference between the feature values, scaled by the difference between the maximum and minimum values of the feature, as shown in equation (2.7) (Daelemans et al., 2004).

$$\delta(x_i, y_i) = \begin{cases} abs(\frac{x_i - y_i}{max_i - min_i}) & \text{if numeric, else} \\ 0 \text{ if } x_i = y_i \\ 1 \text{ if } x_i \neq y_i \end{cases} \tag{2.7}$$

In this way, the degree of similarity between the values is taken into account. However, most feature values used in NLP are symbolic, and for such features, the Overlap metric considered so far simply checks whether two feature values match or not. Still, some values may be more closely related than others in a way that is reflected in their patterns of co-occurrence with categories.

For example, in named entity recognition, words that often occur to the left of the name when the name is a location (e.g., prepositions such as *in* and *through*) can be considered more similar to each other than to words that occur in that position when the name belongs to a different category. The Modified Value Difference Metric, or MVDM (Cost and Salzberg, 1993; Stanfill and Waltz, 1986), computes the distance between two values of a feature in a way that reflects their patterns of co-occurrence with categories, using the following equation (Daelemans et al., 2004)[3]:

$$\delta(v_1, v_2) = \sum_{i=1}^{n} |P(C_i|v_1) - P(C_i|v_2)| \tag{2.8}$$

In fact, as pointed out by Hoste (2005, p. 176), the use of MVDM brings an element of unsupervised learning into MBL, because MVDM can be seen as a form of clustering of feature values into similarity clusters.

[3]TiMBL, the software package that was used for the memory-based learning experiments in this work, also provides another similarity metric, called the *dot-product* metric. However, since this metric was not used in the present experiments, I will not describe it here but rather refer the interested reader to Daelemans et al. (2004).

2.2.4 Jeffrey Divergence Metric

The Jeffrey Divergence Metric works in a way which is similar to that of the MVDM, but it calculates a larger difference between values whose classes have more orthogonal distributions. According to Daelemans et al. (2004), this makes it more robust in the face of sparse data. The metric is calculated as follows:

$$\delta(v_1, v_2) = \sum_{i=1}^{n} (P(C_i|v_1) log \frac{P(C_i|v_1)}{m} + P(C_i|v_2) log \frac{P(C_i|v_2)}{m}) \tag{2.9}$$

$$m = \frac{P(C_i|v_1) + P(C_i|v_2)}{2} \tag{2.10}$$

2.3 Maximum entropy modelling

Maximum entropy (MaxEnt) modelling (Jaynes, 1983) has been applied to a variety of NLP tasks. Below is a list of a few of these tasks, with references to some central work in which maximum entropy modelling has been applied.

- sentence boundary determination (Mikheev, 1998, 2000, 2002)
- machine translation (Berger, Della Pietra, and Della Pietra, 1996)
- part-of-speech tagging (Ratnaparkhi, 1996)
- syntactic parsing (Charniak, 2000; Ratnaparkhi, 1999)
- named entity recognition (Bender, Och, and Ney, 2003; Borthwick, 1999; Chieu and Ng, 2003; Curran and Clark, 2003a; Florian, Ittycheriah, Jing, and Zhang, 2003; Haaland, 2008; Klein, Smarr, Nguyen, and Manning, 2003; Malouf, 2002b; Mikheev, Moens, and Grover, 1999),
- PP attachment disambiguation (Ratnaparkhi, Reynar, and Roukos, 1994)

The basic idea behind this modelling technique is very intuitive. We have some phenomenon (e.g., a language) that we want to model. We create a sample (in the case of language, a corpus) of the phenomenon, which we will use to estimate the parameters of our model. Making the common (though usually inaccurate) assumption that our sample is representative of the phenomenon, we want our model to have the following properties:

- It should reflect the facts in the population from which the sample is drawn as closely as possible[4].

- It should not make any claims that are not supported by the evidence found in the sample.

The first point is shared by most modelling techniques, but the second one is usually not addressed explicitly and may or may not be fulfilled by a particular technique. The MaxEnt modelling technique makes a point of fulfilling this requirement by insisting that, among all possible models that reflect the facts of the sample equally well (i.e., those models that fulfil the first requirement), we should select the one that is maximally agnostic about all other potential facts, i.e., the one that is maximally uncertain about everything not found in the training sample.

Information theory (Shannon, 1948) tells us that uncertainty with respect to information can be expressed as entropy (cf. section 2.2.2). Thus, by selecting the model (among those compatible with the data) that maximizes the entropy, we are choosing the one that goes furthest towards the goal of not making any unwarranted claims about the phenomenon in question. We can maximize the entropy in the model by setting its probability distribution to be as uniform as possible, subject to the constraints imposed by those probabilities that *can* actually be computed from the training sample.

A number of methods have been developed for estimating the parameters of a MaxEnt model. These include, with standard references, Generalized Iterative Scaling (Darroch and Ratcliff, 1972), Improved Iterative Scaling (Berger et al., 1996; Della Pietra, Della Pietra, and Lafferty, 1997), the use of feature lattices (Mikheev, 1998), and application of a hill-climbing method designed for solving nonlinear optimization problems, more specifically the Limited Memory Variable Metric method first presented by Benson and Moré (2001) and later applied to MaxEnt parameter estimation by Malouf (2002a). A commonly used method is the Improved Iterative Scaling (IIS) algorithm, which is laid out as follows by Berger et al. (1996) (who also provide further details about the calculation of the crucial step—2a):

`Input:` Feature functions $f_1, f_2, ...f_n$; empirical distribution $\tilde{p}(x, y)$
`Output:` Optimal parameter values $\lambda_i^\star$; optimal model $p_{\lambda^\star}$

1. Start with $\lambda_i = 0$ for all $i \in \{1, 2, ..., n\}$

2. Do for each $i \in \{1, 2, ..., n\}$:

[4]However, the sample may contain outliers and errors or other kinds of noise that are not true characteristics of the population we want to model, and that will not generalize well to new data. Thus, we should try to avoid overfitting the model to the sample.

(a) Let $\Delta\lambda_i$ be the solution to

$$\sum_{x,y} \tilde{p}(x)p(y|x)f_i(x,y)\exp(\Delta\lambda_i f^{\#}(x,y)) = \tilde{p}(f_i) \tag{2.11}$$

where $f^{\#}(x,y) \equiv \sum_{i=1}^{n} f_i(x,y)$

(b) Update the value of λ_i according to: $\lambda_i \leftarrow \lambda_i + \Delta\lambda_i$

3. Go to step 2 if not all the λ_i have converged

Note that there is a difference in the way that the term *feature* is used in maximum entropy modelling with respect to the way it is used in memory-based learning. In section 2.2.1, this term was used to refer to a single position in the feature vector (e.g., *0* or *cap*) that could take on a range of different values (e.g., *he* or *Washington*). In the context of MaxEnt models, on the other hand, a feature is a combination of some aspect of the data sample with some predicted category (for instance, a particular feature might be paraphrased as "the lemma of the name is *Washington* and its category is PERSON"). In fact, the feature is really a binary-valued function, or *indicator function* (the f_i shown in the IIS algorithm above), that takes on the value 1 or 0 depending on whether or not it matches the current data context *and* its prediction is set to be true.

Once the parameters for the MaxEnt model have been estimated, the probability of a certain category y given a certain context x (e.g., a phrase or a sentence) can be calculated using the following equations (Berger et al., 1996)[5]:

$$p(y|x) = \frac{1}{Z(x)}\exp(\sum_i \lambda_i f_i(x,y)) \tag{2.12}$$

where $Z(x)$ is a normalizing constant that is needed in order to satisfy the requirement that $\sum_y p(y|x) = 1$ for all x, and is given by:

$$Z(x) = \sum_y \exp(\sum_i \lambda_i f_i(x,y)) \tag{2.13}$$

An advantage of MaxEnt models is that they can handle features that are not statistically independent of each other. This means, for instance, that if we are modelling some language phenomenon, we can include in the feature set the occurrence of two specific phenomena in a corpus so that they can each individually contribute to the classification decision, but at the same time we can also include a feature that consists of the *combination* of these features. The individual features and their combination

[5]In fact, Berger et al. use the notation $p_\lambda(y|x)$ and $Z_\lambda(x)$ due to details of their development of the solution to the problem of parameter estimation. I have skipped the lambda subscripts in my rendering of their equations, since they have no significance outside of that context.

are obviously not independent of each other, and should therefore not be used simultaneously in most other types of statistical methods (which assume independence between the features that are used). MaxEnt models, on the other hand, handle this kind of situation gracefully, distributing the weight for the individual features and their combination appropriately based on their occurrences in the corpus.

The number of possibly useful MaxEnt features could run into the thousands or even millions, making the parameter estimation procedure intractable. The problem may be further aggravated by the fact that combinations of individual features may be included, as noted above. This might lead us to look for some principled way of choosing only those features that will contribute usefully to the classifier accuracy while being susceptible to reliable estimation from our training data. One such feature selection algorithm is presented by Berger et al. (1996); an alternative approach using feature lattices is proposed by Mikheev (1998). In practice, however, the feature selection process is often based simply on the intuitions of the experimenter about what would be useful features for the particular domain that should be modelled, along with frequency cutoff thresholds to avoid including features that are too infrequent to be estimated reliably.

2.4 Support vector machines

The use of support vector machines (Burges, 1998; Vapnik, 1995) is a modelling technique that improves on the classification approach taken by perceptron-based neural networks (Rosenblatt, 1958) and back-propagation networks (Rumelhart, Hinton, and Williams, 1986). Like these neural networks, support vector machines (SVMs) aim at finding a hyperplane that separates vectors in a vector space that belong to different categories. However, in general there are an infinite number of hyperplanes that perform the same separation of the space. While they all do the same job with respect to this initial space, different hyperplanes may lead to different classification of any new vectors added to the space if those new vectors lie close to the plane.

The improvement provided by SVMs is that they not only try to find a hyperplane that separates the vectors belonging to two different classes, but they try to find the one hyperplane that is *maximally distant* from all these vectors. The rationale behind this is that by using such a hyperplane, we maximize the chance that any new vectors will fall on the correct side of the hyperplane. In other words, the plane will be the one most likely to assign any new vectors to their most suitable category.

In order to find the desired hyperplane, it is sufficient to maximize the distance to those vectors that are closest to the plane, so we can in fact discard all the other vectors. The remaining vectors, those that were closest to the hyperplane, are called the *support vectors* of the plane.

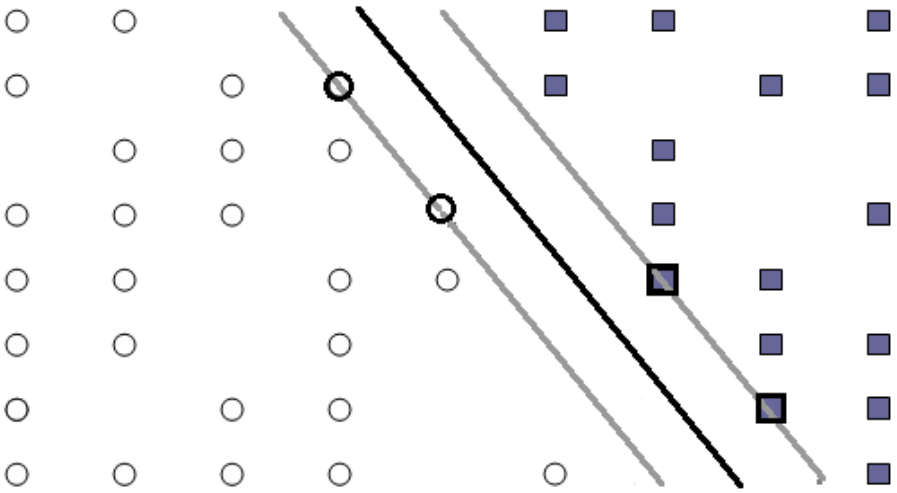

Figure 2.2: Illustration of the idea behind support vector machines. Circles and squares are separated in the vector space by a hyperplane which is maximally distant from each support vector (displayed in bold).

This process is illustrated in Figure 2.2. The support vectors are marked with thick borders. There are three parallel hyperplanes: two that go through the support vectors (the grey ones), and one—the hyperplane that separates the classes—which is maximally distant from each of the other two. The distances between the dividing hyperplane and the other hyperplane to either side are called *margins*, and the goal of the SVM learning algorithm is to maximize these margins.

In addition to showing very good classifier accuracy in a range of modelling tasks, SVMs have the advantage of being flexible with respect to the different *kernels* used in their calculation. Kernels transform the input vector space, with the goal of creating a vector space in which vectors belonging to different categories can be more easily separated.

As an example, consider Figure 2.3, which shows the effect of applying a kind of mirroring kernel to an input space that represents the *exclusive-or* (XOR) problem[6]. XOR is a classical example in the neural networks literature, used for demonstrating the limitations of linear networks (see, e.g., Rumelhart et al. (1986); a related example demonstrating the same point was given in the seminal work by Minsky and Papert (1969, pp. 13–14)).

The vectors in this space contain two elements, which can have values of either 0 or 1. The class of each vector is the result of applying the XOR operator to it. Applying

[6]I thank Christer Johansson for suggesting this example.

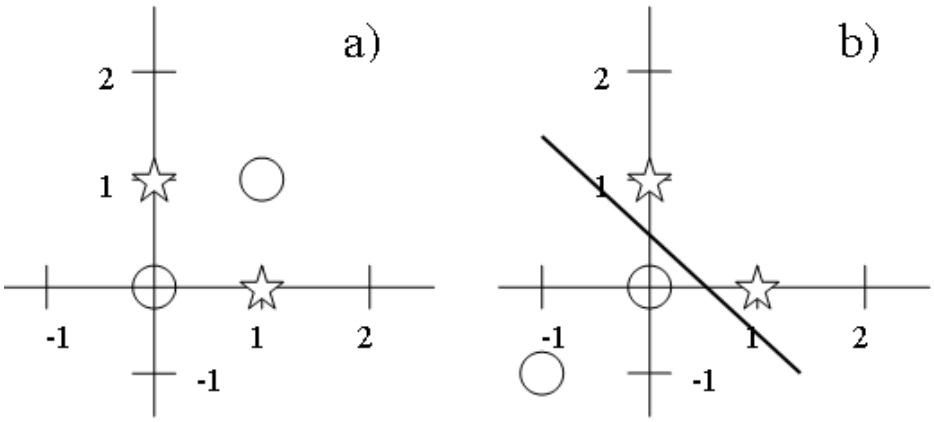

Figure 2.3: Illustration of the effect of a kernel that switches the sign of vector element values if the sum of these values are greater than 1, showing how it transforms a vector space in which the categories are not linearly separable into one in which they are.

the XOR operator to a vector yields TRUE if one, *but not both*, of the elements have a value of 1, and FALSE otherwise. In Figure 2.3, stars represent vectors for which an XOR'ing of the elements yields TRUE, while circles represent vectors for which the result of this operation is FALSE.

Separating the TRUE and FALSE classes in Figure 2.3a cannot be done with a linear classifier; there is no way to draw a straight line through the coordinate system in such a way that each side of the line only contains vectors of a single class. However, we can apply an operation to the vectors that switches the sign of the element values if the sum of these values exceeds 1, as shown in Figure 2.3b. For the vector $[1,\ 1]$, which is represented by the upper right circle in Figure 2.3a, the sum of the elements equals 2, and thus it will be transformed into the vector $[-1,\ -1]$. Hence, the upper right circle will be mirrored around the origo and end up at at the lower left corner, thus making it perfectly possible to draw a straight line that separates the two categories.

Although this is an extremely simplified example which may not seem to have much practical use, it serves to illustrate the so-called "kernel trick" (Aizerman, Braverman, and Rozonoer, 1964) that is used in SVMs, i.e., the effect of applying an operation (a kernel) to the vector space that transforms it from one in which vectors belonging to different classes are not linearly separable into one in which they are.

Different commonly used kernels (e.g., linear, polynomial, or kernels based on Gaussian radial basis functions) provide a trade-off between speed and classifier accuracy. For instance, radial basis functions often provide very good classifier accuracy, but might lead to very long training times when the data set is reasonably large—in fact, it sometimes turns out that the choice of parameter settings that yields the best classification results becomes so computationally expensive as to be practically unusable

when the data set is scaled to a reasonable size. Linear kernels, on the other hand, are fairly efficient, but tend to result in poorer classifier accuracy.

The use of SVMs has yielded very good results on a number of NLP tasks. Some examples of such tasks, and of work that has applied SVMs to these tasks, are:

- shallow semantic parsing (Pradhan, Hacioglu, Krugler, Ward, Martin, and Jurafsky, 2005)
- chunking (Kudo and Matsumoto, 2001)
- dependency structure analysis (Kudo and Matsumoto, 2000)
- named entity recognition (Isozaki and Kazawa, 2002; Mayfield, McNamee, and Piatko, 2003; McNamee and Mayfield, 2002)

In particular, SVMs have good generalization properties, even with very high-dimensional input spaces (Kudo and Matsumoto, 2001), making them less dependent on careful feature selection than other commonly used techniques such as maximum entropy modelling and memory-based learning. The downside to SVMs is that they usually require far more computational resources than these other methods.

The computation of SVMs involves fairly complex mathematics, and since this thesis focuses on the use of machine learning for NLP tasks rather than on the inner workings of various machine learning methods, I will not delve into the details of SVM computation here. Interested readers are referred to Burges (1998) for an introduction to this subject, and to Vapnik (1995) for a more detailed description.

2.5 Advantages of memory-based learning

As mentioned earlier, memory-based learning has been the primary machine learning method used in the present work. In order to justify this choice, it might be useful to give an overview of various advantages that this kind of algorithm holds over alternative, so-called eager algorithms.

2.5.1 Exceptions are not forgotten

Since k-nearest neighbour classifiers, such as those used in memory-based learning, postpone the calculations required to classify an instance until the actual moment of classification, they are often called *lazy* learning algorithms. This is in contrast to *eager* algorithms, which perform the required calculations during the training phase. Most commonly used machine learning algorithms are eager, including maximum entropy modelling and support vector machines.

One advantage of lazy learning algorithms is that since there is no generalization during the training phase, all instances found in the training corpus can have an impact

on the classification of unknown instances. As mentioned in section 2.2, all natural languages contain items that behave more or less idiosyncratically ("all grammars leak", Sapir (1921)), and so NLP systems need to be able to handle idiosyncrasies and sub-regularities involving only a few items in the language. Eager algorithms tend to effectively filter out such phenomena, and thus lazy algorithms hold an advantage in this respect.

Filtering out exceptional instances can even harm the generalization capacity of a system. Daelemans et al. (1999) examine the effect of deleting exceptions from the databases of memory-based systems that are applied to tasks covering a wide spectrum of NLP: phonology and morphology (grapheme-to-phoneme conversion), morphology and syntax (part-of-speech tagging, base noun phrase chunking), and syntax and lexical semantics (prepositional phrase attachment). The authors conclude that deleting exceptions harms generalization accuracy in all of these tasks, a conclusion that runs contrary to a consensus in supervised machine learning that forgetting exceptions is beneficial for generalization (Quinlan, 1993).

Daelemans and van den Bosch (2005) recreate the experiments from Daelemans et al. (1999) and get slightly different results, in which the deletion of exceptions has a less dramatic effect. However, they still show that deleting exceptions is never beneficial in their experiments and that it does become harmful for the generalization accuracy if a large enough portion of exceptions are deleted (Daelemans and van den Bosch, 2005, p. 119).

2.5.2 Local mapping functions

Another advantage of lazy learning, which is related to the previous one, is that classification does not depend on generalization over the entire training corpus (Mitchell, 1997). Eager algorithms attempt to approximate a global mapping function (e.g., from surface forms to named entity categories), meaning that the target function is defined over the entire training corpus. Lazy algorithms, on the other hand, postpone construction of the mapping function until it is time to classify a specific unknown instance. The function is then only formed as a generalization over a small subset of the training instances which are similar to the unknown instance, and this can yield a much more accurate basis for classification.

2.5.3 Leave-one-out testing

Lazy learning makes it practically feasible to do leave-one-out testing of the system (Weiss and Kulikowski, 1991). When doing leave-one-out testing, the entire available manually tagged corpus is read into the database, and then each feature vector in turn is tested against all of the remaining ones. In other words, the entire corpus functions

both as training corpus and as test corpus, maximizing the available training and test data. If we were to use this technique with eager algorithms, we would have to re-train the classifier once per instance in the corpus, which would be an infeasible task for any reasonably sized corpus.

2.5.4 Fast training (but slow application)

Since MBL does not approximate any mapping function during training, the training phase is extremely fast compared to eager methods. Training simply consists of building a database of training instances, with the possible addition of calculating weight values for the various features in the instance vectors (see section 2.2.2).

Because of this, the system can quickly be re-trained if additional training material becomes available. The downside is that testing the system, or applying it to new text for classification purposes, is comparatively slow, since an approximation to a local mapping function must be made for each instance we wish to classify. Because the database must be searched for instances that are similar to each unknown instance, application speed is inversely proportional to the size of the database. The problem of slow application can be somewhat amended by sophisticated methods for indexing the database (Daelemans et al., 2004), but it still remains an important disadvantage, especially for a production system.

Storing all instances in the database also creates a considerable need for storage resources, but with efficient database indexing and the large amount of RAM and disk space available in modern computers, this is rarely a problem. Moreover, implementations of eager algorithms that store the entire training set in memory during training (which is the case, for instance, with Quinlan's C5.0 decision tree implementation; see Daelemans et al. (2004)) in fact require more memory, since both the training instances and the results of generalizations must be stored.

2.5.5 Implicit smoothing

When the features of an unknown instance are identical to the features of one or more existing instances in the database, classification of the unknown instance simply amounts to selecting the category that is shared by the majority of those training instances. However, with most language processing tasks it will frequently be the case that no instance from the database matches the unknown instance exactly. In order to select a category for such instances, it is necessary to implement some kind of smoothing.

For probabilistic methods, one way to do this is to reserve some of the probability mass for unobserved events, which can be done by using a method such as Add-One or (preferably) Good-Turing (Church and Gale, 1991). Another way is to back off

to vectors that contain only a subset of the features that are found in the original (Katz, 1987). Selection of a category is done by interpolation of these less specific vectors. Since the number of less specific vectors grows exponentially with the number of features, to make this computationally feasible for eager algorithms, only certain vectors can be included in the computation. Which vectors to include can be decided based on validation experiments on held-out data (Collins and Brooks, 1995).

With a nearest neighbour approach, we implicitly get the effect of backing off, without the need to calculate which vectors to include (Zavrel and Daelemans, 1997). The vectors used for backing off when classifying a particular unknown instance will be determined by the set of nearest neighbours in the database. For example, Zavrel and Daelemans (1997) show that their MBL system for PP attachment disambiguation automatically backs off to the same vectors as those determined by validation experiments on held-out data conducted by Collins and Brooks (1995). Moreover, computation of the information gain values used in Zavrel and Daelemans' system is many orders of magnitude faster than Collins and Brooks' validation experiments (Zavrel and Daelemans, 1997).

2.5.6 Easy manual inspection

Classifiers that are built on eager learning algorithms approximate the actual mapping function from instances to categories by tuning a set of parameters, thereby compressing the information needed to perform the classification. For example, in hidden Markov models, these parameters take the form of transition and emission probabilities; in maximum entropy models, they are encoded as Lagrange multipliers; and in neural networks, they take the form of synaptic connection weights. As a result, it is very difficult for a human observer to get a clear view of why an instance has been classified in a certain way. It is possible, however, to use techniques such as Cluster Analysis or Principal Component Analysis to reduce the high dimensionality of the parameter space to a two- or three-dimensional space, which can more easily be interpreted by humans.

With MBL, no post-processing of the model is required in order to make it suitable for manual inspection. One can easily produce a list of the nearest neighbours that constitute the support set for a given classification. This means that whenever an instance is given the wrong category, it is possible to study its support set in order to find the basis for the erroneous classification, and perhaps get an idea of types of information that should be added or excluded from the feature vectors in order to enhance the performance of the classifier. If MVDM is used, the picture becomes more complicated, but the results of classification will still be much more suitable for manual analysis than is the case with eager methods.

Chapter 3

Tools and Resources

3.1 Introduction

One of the advantages of data-driven NLP methods over more knowledge-based methods is that they can go beyond the intuitions (and perhaps preconceptions) of the linguist, providing for a much better coverage of the linguistic phenomena in question. The other side of the coin is, of course, that in order to be of any use at all, these methods require data—preferably lots of it. This might be a problem for many languages, especially if the methods require some kind of linguistic preprocessing of the data (e.g., tokenization, lemmatization, part-of-speech tagging, parsing, etc.), since big corpora, and especially linguistically processed ones, are unavailable for many of the languages in the world.

Thus, the use of data-driven methods for NLP requires access to large amounts of electronic text, and in most cases also requires either that the text has already been processed in various ways or that we have access to additional tools that can do the processing for us. At the beginning of this chapter, I will give an overview of the corpus that forms the data material used in this thesis, viz. the Oslo Corpus of Tagged Norwegian Texts, and the linguistic tool that has provided this corpus with various important grammatical information: the Oslo-Bergen tagger.

Although the technical specifications of various machine learning methods that are available in the literature may enable us to implement the methods ourselves, in most cases we would rather use existing implementations if any such implementations exist and are believed to be solid. For all of the methods employed in this work, there are existing implementations that have been developed by researchers with profound knowledge about the underlying mechanisms, and that have been optimized and bug-fixed over an extended period of time. Furthermore, they can all be downloaded from the web and used freely for non-commercial purposes. Consequently, I have chosen to

use these existing tools rather than program the core functionality of the statistical methods myself. At the end of the chapter, I will give a brief presentation of these tools: TiMBL, Zhang Le's maximum entropy modelling toolkit, and SVM^{light}. I will also give a short description of Paramsearch, a tool for automatic pseudo-exhaustive parameter optimization.

3.2 The Oslo Corpus of Tagged Norwegian Texts

There exist a couple of Norwegian corpora which are reasonably large and marked with grammatical information: the Oslo Corpus of Tagged Norwegian Texts[1] (Hagen, Johannessen, and Nøklestad, 2000b), developed and maintained at the Text Laboratory, University of Oslo, and the Norwegian Newspaper Corpus[2] (Hofland, 2000), which is developed at Aksis, University of Bergen. The first of these contains about 24 million words, while the latter consists of approximately 500 million words.

Despite the considerably larger size of the Norwegian Newspaper Corpus, for most of the work described in this thesis I have nevertheless chosen to use the Oslo Corpus. There are several reasons for this choice. Firstly, because it is developed and maintained at my own institution, the Text Laboratory at the University of Oslo, this corpus has been more readily available to me. Secondly, the wider variety of text types available in the Oslo Corpus may result in more robust NLP tools, in the sense that tools trained on more diverse texts are also likely to be able to handle a wider spectrum of texts[3]. Finally, since most of my work employs supervised machine learning, which requires manually annotated material, the amount of data that could be prepared as training material has necessarily been much smaller than the size of the Oslo Corpus. Thus, for most purposes at least, having a bigger corpus would not have helped.

As mentioned above, the Oslo Corpus consists of about 24 million words, which have been morphologically tagged and marked with grammatical dependency relations. The corpus contains texts from several genres[4]: newspapers and magazines, fiction, factual prose, subtitles for television, and unpublished material. It covers both of the Norwegian language varieties, *bokmål* and *nynorsk*, with a major emphasis on *bokmål* (which is the variety that I have chosen to focus on in my work). Table 3.1 provides some information about the *bokmål* part of the corpus.

The corpus contains all of the Norwegian texts that were available to us at the

[1] This corpus has a web-based search interface at `http://www.tekstlab.uio.no/norsk/bokmaal/english.html`.

[2] Web-based search interface at `http://www.tekstlab.uio.no/norsk/bokmaal/english.html`

[3] In fact, the subcorpus used for the actual anaphora resolution task only contains fiction material, since that is the only kind of text that has been annotated in the latest version of the BREDT corpus (see chapter 8). For the named entity recognition and PP attachment disambiguation subtasks, however, a variety of text types were included. See the individual chapters for more information.

[4] The online version only contains newspapers and magazines, fiction, and factual prose, and does not include the 5 million words contributed by the lexicography section at the University of Oslo.

	No. of words	No. of sentences	No. of files
Newspapers	9,691,143	687,163	114
Fiction	6,328,662	490,463	112
Factual prose	7,278,561	407,954	138
TV subtitles	24,975	3774	7
Unpublished	571,582	12,711	8
Total	23,894,923	1,602,065	379

Table 3.1: The *bokmål* part of the Oslo Corpus of Tagged Norwegian Texts.

Text Laboratory at the time it was created (in 1999), but has later been extended by about 5 million words from various genres contributed by the lexicography section at the University of Oslo.

The corpus is not deliberately balanced with respect to genres. While this might be problematic for some types of linguistic research, I do not consider it to be a problem when the corpus is used as data material for the kind of NLP work described here, as long as one is clear about what kind of genres make up the texts used for training and testing the various systems. Most (though not all) of the systems and computations presented in this thesis use only selected parts of the corpus, and details about the kinds of text used will be given as part of the descriptions of the individual systems in the following chapters.

Figure 3.3 on page 58 shows an example of what the Oslo Corpus looks like in its original form, i.e., morphologically and syntactically tagged using the Oslo-Bergen grammatical tagger, with English glosses added to the left. The text that is analyzed is the following:

(1) Tilfeldig møte på Heathrow
"Chance encounter at Heathrow"

HOVEDVEIEN UT MOT flyplassen var støvet. Husene langs ruten lå tilbaketrukket mellom tørre, brunsvidde hageflekker,
"The main road to the airport was dusty. The houses along the way lay receded between dry, scorched garden patches,"

In the figure, each Norwegian word form is shown left-justified, followed by one or more indented lines displaying the reading or set of readings that remain after disambiguation (see section 3.3 for more information about the disambiguation process). Each reading contains the lemma form of the word given in quotes, followed by part-of-speech and other morphological information, and finally one or more syntactic categories marked by an at-sign ("commercial at").

This format was used for some of the computations presented in the current work. However, each of the subtasks that make up the overall system also required the corpus markup to be extended or converted to another format.

For the main task, i.e., anaphora resolution (chapter 8), the data were represented in XML. As part of the work carried out in the BREDT project at the University of Bergen, selected parts of the grammatical information in the original version of the corpus were extracted and used to create a set of reduced grammatical tags[5]. Furthermore, lists of antecedents were manually determined for all anaphors in a small part of the corpus[6]. The annotators carried out their work in accordance with the annotation scheme outlined in Borthen (2004).

Combining the reduced tags with the information in the antecedent lists, I created a version of the corpus in the XML format shown in Figure 3.4 on page 59. The figure shows the following text sequence:

(2) David, sa hun lavt, til seg selv, nesten som et spørsmål. Han er på kjøkkenet, sa Sonja
"David, she said softly to herself, almost like a question. He is in the kitchen, said Sonja"

The subcorpus marked up in this way was used to train and test the anaphora resolution system described in chapter 8.

Because the anaphora resolution system works with unicode, the text is encoded in UTF-8. Note that each token has been given an ID. An anaphor-antecedent relationship is represented by an <ant> element functioning as a child of the token (<ord>) element representing the anaphor, and the <ant> element contains the ID of the antecedent.

For example, in the figure, the token *seg* "himself/herself/itself/themselves" with ID=5235 is marked as having an antecedent with ID=5231 (i.e., *hun* "she"), and the anaphor-antecedent relation is marked as being of type *r*, i.e., the **coreference** type (which is in fact the only relation type handled by the anaphora resolution system presented in this thesis). Similarly, han_{5244} is linked to $David_{5228}$, which is further linked to a sequence of words (an NP) that occurs earlier in the text.

For named entity recognition (chapter 4), the only required modification to the original format was to insert additional tags that represent the named entity categories of proper names, as illustrated in Figure 3.5 on page 60. The snippet displayed is the following:

[5]This extraction and conversion work was carried out by Christer Johansson and Lars G. Johnsen.
[6]In my work, I have used the second version of the data, which was annotated by Kaja Borthen and Lars G. Johnsen.

(3) Tyrkia har satt Norge på sin røde liste, og Penguin til Tyrkia er nå et dødt prosjekt, fastslår Jørgensen.
"Turkey has put Norway on its red list, and Penguin to Turkey is now a dead project, states Jørgensen."

Named entity tags are shown in bold in the figure. They start with an ampersand and end with an asterisk; the figure shows examples of four of the named entity categories I have used in my work: PERSON (&pe*), ORGANIZATION (&or*), LOCATION (&st*), and OTHER (&an*) (the WORK and EVENT categories are not exemplified).

Finally, the PP attachment disambiguation system (chapter 5) uses the original format for training, but the development and test corpora have been created in HTML with noun and verb attachment marked using different cascading style sheet (CSS) classes, as shown in Figure 3.6 on page 61. CSS is the standard technology used to style the contents of a web page, e.g., with different text colours, font sizes, etc. Through its use of CSS, the format that is employed here makes it easy to display noun and verb attachment differently (e.g., using different colours), for easy inspection in a web browser. Hence, it facilitates manual work with PP attachments considerably.

More details about the task-specific versions of the corpus are provided in the respective chapters.

3.3 The Oslo-Bergen tagger

The Oslo Corpus has been tokenized, lemmatized, and morphologically tagged and marked with syntactic dependency relations using the Oslo-Bergen tagger (Hagen, Johannessen, and Nøklestad, 2000a)[7]. Both morphological and syntactic tagging are based on the Constraint Grammar formalism (Karlsson, Voutilainen, Heikkilä, and Anttila, 1995) in combination with the *Norsk ordbank* lexical database and morphological analyzer[8] as well as the analyzer for compounds and unknown words described by Johannessen and Hauglin (1998).

Constraint Grammar (CG) is a disambiguation framework in which a grammar is expressed as a set of hand-written disambiguation rules. The rule format is illustrated in Figure 3.1. Note that although several improved versions of the CG formalism have been presented over the last decade (CG2: Tapanainen (1996); VISLCG, which was

[7] The lexical database and the morphological analyzer have been developed jointly by the Text Laboratory and the Documentation project at the University of Oslo, using material from *Bokmålsordboka* and *Nynorskordboka* provided by the lexicography section at the same university. The morphological disambiguation rules were written by Kristin Hagen, Janne Bondi Johannessen, and myself. I wrote the original set of syntactic mapping and disambiguation rules, which was later extended by Lilja Øvrelid and Kristin Hagen. The current version of the tagger was implemented in Allegro Common Lisp by Paul Meurer at the University of Bergen. The tagger can be tested online at `http://decentius.hit.uib.no:8005/cl/cgp/test.html` (interface in Norwegian).

[8] *Norsk ordbank* can be searched online at `http://www.dok.hf.uio.no/perl/search/search.cgi` (interface in Norwegian).

```
(@w =0 (subst ent ub)
    (NOT 0 adverbialsubst)
    (NOT 0 tall-subst)
    (NOT 0 målsubst)
    (NOT 0 gen)
    (NOT 0 tittel)
    (NOT 1 komma/konj)
    (NOT 1 anf)
    (NOT -1 det/adj/gen/prop)
    (NOT -1 %hvilken%)
    (NOT -1 prep)
    (NOT -1 komma/konj)
    (*-1 setn-gr *R)
    (NOT *R kop)
    (*1 ikke-adv-adj-det *L)
    (NOT *L setn-gr)
    (*1 ikke-adv-adj-det *L)
    (NOT *L fork/ukjent-ord)
    (*1 ikke-adv-adj-det *L)
    (NOT *L parentes)
    (*1 ikke-adv-adj-det *R)
    (LR0 subst *R)
    (NOT LR0 ikke-subst *R)
    (NOT LR0 adverbialsubst *R)
    (NOT LR0 subst-prop *R)
    (NOT LR0 fork))
```

Figure 3.1: Example of a morphological disambiguation rule which removes the "singular indefinite noun" reading from a word if the given specification of the word context applies. See Karlsson et al. (1995) for details about the rule format.

developed by Martin Carlsen; and CG3, developed by Tino Didriksen[9]), the Oslo-Bergen tagger employs the original version that was presented by Karlsson et al. (1995), since that was the only version available at the time the work on the tagger began[10].

Before disambiguation, a morphological analyzer provides all possible grammatical readings of a word, and then the rules are applied in order to disambiguate the word. The idea behind this approach is to remove as much ambiguity as possible, while at the same time leaving a word (partially) ambiguous if there is insufficient information in the context to fully disambiguate it. Thus, if the category of a word cannot be fully determined, it is partially disambiguated rather than being left without any analysis at all.

The hand-written rules in a CG grammar can either select a particular reading or just remove one or more unwanted readings. Because of the ambiguity reduction ap-

[9] `http://beta.visl.sdu.dk/cg3/vislcg3.pdf`.

[10] The rules of the Oslo-Bergen tagger are currently being translated into the CG3 formalism as part of the ongoing work in the *Norsk aviskorpus* project.

proach described above, the framework focuses on rules that remove rather than select readings. In most cases, several rules apply to each word, gradually reducing its ambiguity. As a consequence, a CG grammar that is equipped to handle unrestricted text typically consists of a large number of rules. For morphological disambiguation (i.e., disambiguation of part-of-speech along with morphological features), the Norwegian CG grammar for *bokmål* consists of 1440 rules.

In addition to the morphological disambiguation component, the Oslo-Bergen tagger contains a syntactic component, which carries out a kind of syntactic dependency tagging which is specified by the Constraint Grammar formalism. The syntactic tagging process begins with a mapping step that maps each morphological reading that remains after morphological disambiguation to a set of possible syntactic categories. This means that each individual word gets a syntactic tag; there is no phrase level in Constraint Grammar syntax. The mapping step is followed by a disambiguation step in which the word is syntactically disambiguated using a set of rules which are similar in form to those used for morphological disambiguation, as illustrated in Figure 3.2. The *bokmål* syntactic component contains 395 mapping rules and 978 disambiguation rules.

```
(@w =s0 (@i-obj)
    (0 subst-prop)
    (1 subst-prop)
    (NOT *1 overskrift)
    (*1 ikke-subst-prop *R)
    (LR0 @iv L-1)
    (*-1C s#setn-gr *R)
    (NOT LR0 clb *R)
    (NOT LR0 %som% *R)
    (NOT LR0 @subj *R)
    (NOT *R @subj)
    (*-1C s#setn-gr *R)
    (*R @fv *R)
    (NOT LR0 imp *R)
    (NOT LR0 passiv *R)
    (LR0 kh#trans-med-nom-u-ditrans *R)
    (NOT *R ikke-mod-u-@kon-@<p-utfyll-@adv-@app))
```

Figure 3.2: Example of a syntactic disambiguation rule which removes the "indirect object" reading from a word if the given specification of the word context applies. See Karlsson et al. (1995) for details about the rule format.

3.4 TiMBL

All experiments with memory-based learning (cf. section 2.2) described here were carried out using the Tilburg Memory-based Learner, or TiMBL (Daelemans et al.,

2004)[11]. This software package is developed and maintained by the Induction of Linguistic Knowledge group at Tilburg University, and is a very efficient and feature-rich implementation. It can be run either as a "one-shot" command line classifier or as a server, and there is also a C++ API that makes it possible to integrate the memory-based learning functionality into other programs[12].

TiMBL can handle a number of different input formats, and it recognizes a wealth of command line options that influence the learning algorithm as well as the kind of output provided by the program. Some of the options that TiMBL offers are the various distance metrics and feature-weighting schemes that were presented in section 2.2; for further details, see Daelemans et al. (2004).

3.5 Zhang Le's maximum entropy modelling toolkit

The toolkit used for maximum entropy modelling (cf. section 2.3) in the present experiments was developed by Zhang Le and made available under the Gnu Lesser Public Licence (GLPL)[13]. It is written in C++, and originally surfaced as a fast reimplementation of the maximum entropy package in Java that is part of the OpenNLP project[14].

The toolkit offers implementations of Improved Iterative Scaling (Berger et al., 1996; Della Pietra et al., 1997) and the Limited Memory BFGS Method (Benson and Moré, 2001; Malouf, 2002a), as well as an improved version of the Generalized Iterative Scaling algorithm (Darroch and Ratcliff, 1972) proposed by Curran and Clark (2003b). It also provides a binding to the Python programming language for easy integration into Python programs.

3.6 SVM^{light}

My experiments with support vector machines (cf. section 2.4) make use of the SVM^{light} program by Thorsten Joachims (Joachims, 1999). This program is free for scientific, non-commercial use[15]. It is an efficient C implementation of SVM learning

[11] It should be pointed out that, while TiMBL is by far the most widely known implementation of memory-based learning, it is not the only one. A recent, alternative implementation has been created at the University of Bergen in Norway. This implementation, called BUMBLe (Johansson and Johnsen, 2006), differs from TiMBL in that it takes into consideration the entire instance base rather than a restricted neighbourhood. Since BUMBLe is still under development, it has not been used in the experiments presented here, but in the future it might provide material for interesting comparative studies of the different memory-based approaches taken by these two software packages.

[12] The software package can be freely downloaded for research and education purposes at `http://ilk.uvt.nl/timbl/`.

[13] The toolkit can be downloaded from `http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html`

[14] `http://maxent.sourceforge.net/`.

[15] SVM^{light} can be downloaded from `http://svmlight.joachims.org/`

which can handle several hundred thousands of training examples and thousands of support vectors. SVM^{light} lets users experiment with a large number of parameter settings, and even allows them to define their own kernel functions.

In addition to solving the usual classification and regression problems handled by regular SVMs, SVM^{light} supports an algorithm called SVM^{struct} which can be used to predict complex outputs like trees or sequences, for instance for natural language parsing.

3.7 Paramsearch

The Paramsearch program, developed by Antal van den Bosch, can be used to optimize the most important parameter settings of a range of machine learning tools, including TiMBL, Zhang Le's maximum entropy modelling toolkit, and SVM^{light}. Each of these tools requires a set of parameter values to be specified, and van den Bosch (2004) shows that the performance of the tools is heavily influenced by the choice of parameter values. Van den Bosch points out that learning mechanisms that require many parameters to be set (such as memory-based learning) are likely to benefit more from parameter optimization than those that take fewer parameters (such as maximum entropy modelling). The work by Hoste (2005) shows that parameter optimization does indeed have a major impact on the performance of machine learning methods in coreference resolution.

When run on a data set of more than 1000 items, the Paramsearch program uses a technique called *wrapped progressive sampling* (WPS) (van den Bosch, 2004) to make it feasible to do a pseudo-exhaustive search for optimal parameter settings (if the number of items is 1000 or less, 10-fold cross-validation is used).

The WPS algorithm starts by taking a small part of the available data, dividing it into a training part (80%) and a test part (20%), and testing a wide range of combinations of parameter values on this data. Based on this first run, only the best performing combinations are kept, and the size of the training and test material made available to the classifier is increased. This procedure continues for several iterations, at each step keeping only the best-performing settings and increasing the material. The algorithm stops when either a) a single best-performing combination is found, or b) the entire available training and test material has been used. In the latter case, the default settings of the algorithm are selected if they constitute one of the winning combinations; otherwise, one of the winning combinations is chosen at random.

3.8 Evaluation

For evaluation of the systems developed in this thesis, I have employed four commonly used measures: *accuracy*, *recall*, *precision*, and *F-score*.

Accuracy is a measure of the overall proportion of correct outputs made by a system. It is defined in the following way:

$$\text{accuracy} = \frac{\text{number of correct outputs}}{\text{total number of outputs}} \tag{3.1}$$

The accuracy measure is useful for calculating the overall performance of a system, but it will not tell us whether a classifier shows different performance on different classes of items in the test material. This can instead be measured by recall, precision, and F-score (Rijsbergen, 1979), which are defined as follows:

$$\text{recall for class C} = \frac{\text{number of correctly identified items in class C}}{\text{total number of items in class C}} \tag{3.2}$$

$$\text{precision for class C} = \frac{\text{number of correctly identified items in class C}}{\text{total number of items assigned to class C by the classifier}} \tag{3.3}$$

$$F_{\beta=1} = \frac{2 * \text{recall} * \text{precision}}{\text{recall} + \text{precision}} \tag{3.4}$$

The F-score takes a β parameter that determines how recall and precision are weighted relative to each other. The most commonly used value of β is 1, which means that they are weighted equally; this is the case shown in equation (3.4) and used in all F-score calculations in this thesis.

All tests for statistical significance are carried out using McNemar's test, which is commonly applied in work on language technology. The test is used to make pairwise comparisons of classifiers that are applied to the same test data. The input to the test is the number of unique errors made by each classifier (i.e., errors made by one classifier but not the other, and vice versa). The null hypothesis is that unique errors made by one classifier are equally likely as unique errors made by the other classifier, and the test statistic calculates the probability of seeing the observed error difference given this null hypothesis. If the probability is sufficiently low, we reject the null hypothesis and conclude that there is a significant difference between the classifiers. Unless stated otherwise, whenever the word "significant" is used in this thesis, it implies statistical

significance at the 1% level (i.e., $p \leq 0.01$).

On the following pages, we conclude this chapter with the corpus format examples for various subtasks that were discussed in section 3.2. Then, in the next chapter, we turn to the description of the first support module of the anaphora resolution system, i.e., the named entity recognizer.

```
chance          "<Tilfeldig>"
                    "tilfeldig" adj pos noeyt ub ent @adj>
encounter       "<møte>"
                    "møte" subst noeyt appell ub ent @loes-np
at              "<på>"
                    "på" prep @adv
Heathrow        "<Heathrow>"
                    "Heathrow" subst prop @<p-utfyll
                "<|>"
                    "$|" clb <overskrift> <<<
the-main-road   "<HOVEDVEIEN>"
                    "hovedvei" subst mask appell be ent @subj
out             "<UT>"
                    "ut" prep @adv
towards         "<MOT>"
                    "mot" prep @adv
the-airport     "<flyplassen>"
                    "flyplass" subst mask appell be ent @<p-utfyll
was             "<var>"
                    "være" verb pret a5 pr1 pr2 <aux1/perf_part> @fv
dusty           "<støvet>"
                    "støve" verb perf-part i2 pa1 n pa4 @iv
                "<.>"
                    "$." clb <punkt> <<<
the-houses      "<Husene>"
                    "hus" subst noeyt appell be fl @subj
along           "<langs>"
                    "langs" prep @adv
the-way         "<ruten>"
                    "rute" subst mask appell be ent @<p-utfyll
lay             "<lå>"
                    "ligge" verb pret i2 tr11 pa4 a5 @fv
receded         "<tilbaketrukket>"
                    "tilbaketrukket" adj pos noeyt ub ent @adv
between         "<mellom>"
                    "mellom" prep @adv
dry             "<tørre>"
                    "tørr" adj pos fl @adj>
                "<,>"
                    "$," <komma>
scorched        "<brunsvidde>"
                    "brunsvidd" adj pos fl @adj>
garden-patches  "<hageflekker>"
                    "hageflekk" subst mask appell ub fl @<p-utfyll
                "<,>"
                    "$," clb <komma> @s-gr
                    "$," <komma>
```

Figure 3.3: Excerpt from the Oslo Corpus of Tagged Norwegian Texts in the format produced by the Oslo-Bergen tagger, as discussed in section 3.2. English glosses in bold have been added.

```
David       <ord ne="pe" suff="vid" id="5228" genus="m" lemma="David" syn="np"
                  pos="prp"><ant type="r" id="5187-5188"/>
                  David</ord>
,           <ord suff="," id="5229" lemma="$," syn="sgr" pos="cma">
                  ,</ord>
said        <ord suff="sa" id="5230" lemma="si" syn="fv" pos="v">
                  sa</ord>
she         <ord suff="hun" id="5231" num="sg" genus="f" lemma="hun" syn="subj"
                  pos="pron"><ant type="r" id="5213"/>
                  hun</ord>
softly      <ord suff="avt" id="5232" num="sg" genus="n" lemma="lav" syn="adv"
                  pos="a">
                  lavt</ord>
,           <ord suff="," id="5233" lemma="$," syn="unk" pos="cma">
                  ,</ord>
to          <ord suff="til" id="5234" lemma="til" syn="adv" pos="p">
                  til</ord>
himself/herself/itself/themselves
            <ord suff="seg" id="5235" lemma="seg" syn="pfill" pos="pron">
                  <ant type="r" id="5231"/>
                  seg</ord>
self        <ord suff="elv" id="5236" lemma="selv" syn="det" pos="d">
                  selv</ord>
,           <ord suff="," id="5237" lemma="$," syn="unk" pos="cma">
                  ,</ord>
almost      <ord suff="ten" id="5238" lemma="nesten" syn="adv"
                  pos="adv">
                  nesten</ord>
like        <ord suff="som" id="5239" lemma="som" syn="adv" pos="p">
                  som</ord>
a           <ord suff="et" id="5240" num="sg" genus="n" lemma="en" syn="det"
                  pos="d">
                  et</ord>
question    <ord suff="mål" id="5241" num="sg" genus="n" lemma="spørsmål"
                  syn="pfill" pos="ni">
                  spørsmål</ord>
.           <ord suff="." id="5242" lemma="$." syn="unk" pos="dot">
                  .</ord>
-           <ord suff="xsep" id="5243" lemma="$-" syn="unk" pos="sep">
                  -</ord>
he          <ord suff="han" id="5244" num="sg" genus="m" lemma="han" syn="subj"
                  pos="pron"><ant type="r" id="5228"/>
                  Han</ord>
is          <ord suff="er" id="5245" lemma="være" syn="fv" pos="v">
                  er</ord>
on          <ord suff="på" id="5246" lemma="på" syn="adv" pos="p">
                  på</ord>
the-kitchen <ord suff="net" id="5247" num="sg" genus="n" lemma="kjøkken"
                  syn="pfill" pos="nd"><ant type="r" id="5022"/>
                  kjøkkenet</ord>
,           <ord suff="," id="5248" lemma="$," syn="sgr" pos="cma">
                  ,</ord>
said        <ord suff="sa" id="5249" lemma="si" syn="fv" pos="v">
                  sa</ord>
Sonja       <ord ne="pe" suff="nja" id="5250" genus="f" lemma="Sonja" syn="subj"
                  pos="prp"><ant type="r" id="5205"/>
                  Sonja</ord>
```

Figure 3.4: An example of the XML format used in the anaphora resolution task described in chapter 7, as discussed in section 3.2. English glosses in bold have been added.

```
Turkey      "<Tyrkia>"
                "Tyrkia" subst prop &or*
has         "<har>"
                "ha" verb pres pa1 a6 d5 rl6 tr6 d6/til pa3 tr12 <aux1/perf_part> pa6
put         "<satt>"
                "sette" verb perf-part tr1 rl4 pa1 a6 rl6 pa1/til pa2 pa5 pa3 a9 tr23
Norway      "<Norge>"
                "Norge" subst prop &or*
on          "<på>"
                "på" prep
its         "<sin>"
                "sin" det poss mask ent
red         "<røde>"
                "rød" adj pos be ent
list        "<liste>"
                "liste" subst mask appell ub ent
            "<,>"
                "$," <komma>
and         "<og>"
                "og" konj clb
Penguin     "<Penguin>"
                "Penguin" subst prop &an*
to          "<til>"
                "til" prep
Turkey      "<Tyrkia>"
                "Tyrkia" subst prop &st*
is          "<er>"
                "være" verb pres a5 pr1 pr2 <aux1/perf_part>
now         "<nå>"
                "nå" adv
a           "<et>"
                "en" det kvant noeyt ent
dead        "<dødt>"
                "død" adj pos noeyt ub ent
project     "<prosjekt>"
                "prosjekt" subst noeyt appell ub ent
            "<,>"
                "$," clb <komma>
states      "<fastslår>"
                "fastslå" verb pres tr1 tr2
Jørgensen   "<Jørgensen>"
                "Jørgensen" subst prop &pe*
            "<.>"
                "$." clb <punkt> <<<
```

Figure 3.5: Excerpt from the portion of the Oslo Corpus that has been marked with named entity tags, as discussed in section 3.2. English glosses in bold have been added.

- We have seen a decrease in public support of 8-9 percent this year.
- Vi har hatt en nedgang i publikumsoppslutningen på 8-9 prosent det siste året .

When I agreed to a new period at the meeting in 1993,
Da jeg på kretstinget i 1993 sa ja til en ny periode ,

In the middle of the month there will be two downhill races in Cortina.
I midten av måneden skal det kjøres to utforrenn i Cortina .

This is the first time this boy from Gjovik has advanced to the podium in a single race in the World Cup.
Det er første gang Gjøvik-gutten har tatt steget opp+på seierspallen i et enkeltrenn i verdenscupen .

So far, coming third in the combination in Chamonix just before the Olympics last year has been his best achievement in the World Cup.
Fra+før var tredjeplassen i kombinasjonen i Chamonix like før OL i_fjor hans beste verdenscupplassering .

said Harald Christian Strand Nilsen til NTB etter pressekonferansen.
sa Harald Christian Strand Nilsen til NTB etter pressekonferansen .

when I started the second run today,
da jeg startet andreomgangen i dag ,

Now the two top clubs will meet in Parma tomorrow.
Nå møtes de to toppklubbene i Parma i_morgen .

but have a delayed match (against Milan).
men har en kamp til gode (mot Milan) .

Figure 3.6: The beginning of the development corpus used for PP attachment disambiguation, showing the HTML format that was used for this task, as discussed in section 3.2. English translations in bold have been added.

Chapter 4

Named Entity Recognition

4.1 Introduction

The aim of named entity recognition (NER) is to perform a semantic classification of proper names, and sometimes also certain other kinds of *named entities* such as dates and monetary amounts. In my own work, I have focused exclusively on proper names, and hence the current chapter will only be concerned with this kind of named entity. I have presented a preliminary version of my NER system in Nøklestad (2004); in this chapter I give a more comprehensive description of the system and also report on a number of additional optimization experiments using Paramsearch (van den Bosch, 2004).

The NER process consists of two main stages: a) identification of the word or string of words that constitutes the named entity, and b) selection of the named entity category. This chapter presents a system that performs both tasks. The first task is solved by a morphosyntactic tagger—the Oslo-Bergen tagger, described in section 3.3—by extending it with the capability to recognize word sequences that constitute a proper name. This task has been carried out by other participants in the Nomen Nescio project (Hagen, 2003; Johannessen, Meurer, and Hagen, 2003). My contribution to the system has consisted of the development of data-driven methods to perform the second task, i.e., classification of the named entities into semantic categories.

Most speakers of a certain language, or, more generally, people that are familiar with the culture in which that language is spoken, would probably be able to see that *some* names in the language are ambiguous between different categories. For instance, in the Western world it is widely known that the name *Philip Morris* might denote a person or a company, and many Americans and people that are familiar with American culture would agree that a name like *The Washington Post* might refer to either the newspaper itself or to the organization that publishes it.

However, since new names appear all the time (especially company names and names of books and movies, etc.), and since a name that used to be unambiguous (e.g., a family name) may suddenly become ambiguous (e.g., if it also becomes a company name), all proper names are potentially ambiguous. Thus, rather than focusing on certain names that are known to be ambiguous, in this work I will assume that *every* occurrence of *any* proper name needs to be classified with respect to a set of named entity categories.

4.2 Relevance for anaphora resolution

The classification of named entities is important for anaphora resolution due to the fact that some pronouns (such as *he* and *she*) usually refer to human beings, while others (such as *it*) typically refer to non-human entities. Hence, it is vitally important to know whether a noun denotes a human being, since this information strongly influences which pronouns the noun can corefer with.

In the case in which the noun does indeed denote a human being, it is also useful to know the gender of that person. The gender of many common Norwegian first names is given in the lexicon used by the Oslo-Bergen tagger; thus, if it is decided that a certain name denotes a person, the gender of that person can in many cases be found in the lexicon. Even when the gender cannot be determined, however, knowledge about animacy alone helps tremendously in narrowing down the set of possible antecedents.

In chapter 6, I present techniques for extracting animate nouns from large text collections and for estimating the animacy value of nouns from such collections. I describe a so-called off-line technique in which I harvest a large number of animate nouns from the World Wide Web. Additionally, a (pseudo-)online approach is described, which evaluates the animacy of a given noun as it is encountered in a text. The current chapter describes a different approach, in which *proper names* are classified into one of a number of pre-defined classes, one of which is the PERSON class.

Compared to the techniques presented in chapter 6, this approach is more limited in the type of nouns it applies to, since it is only used for proper names. On the other hand, named entity recognition can be more generally applied to *all* proper names in a text, meaning that the animacy information obtained is not limited to certain lists of nouns like the ones produced by the offline technique in chapter 6. Furthermore, unlike the (pseudo-)online technique described in chapter 6, named entity recognition avoids the need to run hundreds or thousands of queries per noun in order to determine the animacy value of the noun. I hypothesize that both approaches can provide useful input to an AR system, and this will be tested empirically in chapter 8.

4.3 Earlier work

A range of machine learning approaches have been applied to the task of named entity recognition in the past. Most systems have employed eager algorithms (cf. section 2.5.1). Below is a list of earlier work, organized according to the main method that has been used in each case.

- hidden Markov models: Bikel, Miller, Schwartz, and Weischedel (1997); Burger, Henderson, and Morgan (2002); Florian et al. (2003); Jansche (2002); Klein et al. (2003); Malouf (2002b); Mayfield et al. (2003); Whitelaw and Patrick (2003); Zhou and Su (2002)
- decision trees: Black and Vasilakopoulos (2002); Sekine, Grishman, and Shinnou (1998)
- boosting: Carreras, Màrquez, and Padró (2003b); Carreras, Márques, and Padró (2002); Wu, Ngai, and Carpuat (2003); Wu, Ngai, Carpuat, Larsen, and Yang (2002)
- maximum entropy models: Bender et al. (2003); Borthwick (1999); Borthwick, Sterling, Agichtein, and Grishman (1998); Chieu and Ng (2003); Curran and Clark (2003a); Florian et al. (2003); Haaland (2008); Kazama (2004); Klein et al. (2003); Malouf (2002a); Mikheev, Grover, and Moens (1998); Mikheev et al. (1999)
- support vector machines: Isozaki and Kazawa (2002); Mayfield et al. (2003); McNamee and Mayfield (2002); Shen, Zhang, Su, Zhou, and Tan (2004)
- neural networks: Carreras, Màrquez, and Padró (2003a); Florian et al. (2003); Hammerton (2003); Zhang and Johnson (2003)
- transformation-based learning: Florian et al. (2003)
- conditional random fields: McCallum and Li (2003); Okanohara, Miyao, Tsuruoka, and Tsujii (2006)

Additionally, Minkov, Wang, and Cohen (2005) use hidden Markov models and conditional random fields to extract (but not classify) named entities from informal texts such as emails and bulletin board messages. All of these methods have been used in well-performing systems.

Attempts at applying memory-based learning to named entity recognition have so far not been able to compete with the best of the systems that use eager methods. Tjong Kim Sang (2002b) applied memory-based learning to NER in Dutch and Spanish as part of the CoNLL-2002 shared task (Tjong Kim Sang, 2002a). Tjong Kim Sang's

MBL-based classifier obtained an $F_{\beta=1}$ score of 75.8 for Spanish and 70.7 for Dutch. These results were fairly average among the scores reached by the systems participating in the task, which ranged from 61.0 to 81.4 for Spanish and from 49.8 to 77.1 for Dutch.

CoNLL-2003 contained the same shared task as the previous year, though this time applied to English and German, and it included two new memory-based systems: Meulder and Daelemans (2003) achieved $F_{\beta=1}$ scores of 77.0 on English and 57.3 on German, while Hendrickx and van den Bosch (2003) obtained $F_{\beta=1}$ scores of 78.2 on English and 63.0 on German. Both of these systems were beaten by most of their competitors.

Although memory-based learning has so far achieved only modest performance on the task of named entity recognition, the fact that this learning method has obtained good results on a range of other linguistic tasks gives us reason to suspect that it might nevertheless perform well on the NER task given the right circumstances. Therefore, I have systematically studied the effect of a number of input features that are typically used in named entity recognition to see whether they improve or impede performance, and, for those features that turn out to be beneficial, to measure the degree of improvement they provide.

Additionally, the system presented here differs from the systems cited above in that it uses a hybrid approach in which proper names are first identified by a rule-based name finder, and then classified by the memory-based learner. By letting the learner focus on words that are known to be in the target domain (i.e., proper names), I hope to improve its performance.

The hypothesis that a more focused target group benefits the system is supported by various other experiments. For example, Hoste (2005) found that, at least in some cases, using different classifiers for pronouns, proper nouns, and common nouns was beneficial for memory-based coreference resolution. In my own anaphora resolution experiments, reported in chapter 8, I found that going even further and using separate classifiers for different groups of pronouns yielded significantly better results than using a single classifier for all pronouns, at least for the *den* pronoun. Another example is provided by word sense disambiguation experiments with memory-based learning, which have achieved good results by employing one classifier for each ambiguous word (see, e.g., Decadt et al., 2004; Hoste et al., 2002; Mihalcea, 2002; Veenstra et al., 2000).

Finally, in order to evaluate the performance of the lazy learner versus an eager method, I compare it to the performance of a maximum entropy model trained on the same data and using the same feature set. As will be demonstrated in later sections, the memory-based learner performs quite well on the task of Norwegian named entity recognition and outperforms the maximum entropy model, showing that, under the right circumstances, memory-based learning can indeed compete with eager methods on this task.

4.3.1 Scandinavian NER: The Nomen Nescio network

I started the work described in this chapter in the context of the Nomen Nescio network (Johannessen et al., 2005). In this network, six NER systems for Swedish, Danish, and Norwegian were developed, including an early version of the memory-based system that is the focus of this chapter. These systems were:

- For Swedish: a system built on context-sensitive finite-state grammars (Kokkinakis, 2001, 2002, 2003, 2004)
- For Danish: systems based on Constraint Grammar (Bick, 2004) and context-sensitive finite-state grammars (Haltrup, 2003)
- For Norwegian: systems based on Constraint Grammar (Jónsdóttir, 2003), maximum entropy modelling (Haaland, 2008), and memory-based learning (my own work, described in Nøklestad (2004) and in the present chapter)

	Recall	Precision	$F_{\beta=1}$
Danish CG	95.0	95.0	95.0
Swedish FS	87.0	94.0	90.4
Norwegian MBL	83.0	83.0	83.0
Norwegian ME	76.0	76.0	76.0
Norwegian CG	96.5	38.4	54.9
Danish FS	Not calculated		

Table 4.1: Recall, precision, and F-scores for the NER systems developed in the Nomen Nescio network. Adapted from Johannessen et al. (2005).

In Johannessen et al. (2005), we evaluated the various systems. As shown in Table 4.1, the Swedish and Danish systems obtain higher performance scores than the Norwegian ones[1]. It should be pointed out, however, that these systems actually try to solve rather different tasks. All of the Norwegian systems try to determine the correct category of a name *depending on its context* (labelled a *Function over Form* strategy by Johannessen et al. (2005)). Thus, if *USA* occurs as a location (a country in the purely geographical sense) in one context and as an organization (a nation) in another, the Norwegian systems try to capture this difference. The Danish and Swedish systems, on the other hand, always classify a particular word form into the same category (i.e., a *Form over Function* strategy), meaning that they would classify *USA* as a location regardless of whether it functions as a geographical location or as an organization in any given context.

[1] The result given for the memory-based system in Johannessen et al. (2005) is the 10-fold cross-validation F-score obtained with manually optimized parameter settings. Because of minor changes to the system after the paper was published, the F-score reported in the paper is somewhat lower than the one given at the bottom of Table 4.10.

It is possible to get quite far with the *Form over Function* strategy using only gazetteers (i.e., lists of names known to belong to certain categories), at least in restricted domains, or even in a more general news text setting if the lists are continuously updated with new names that occur in the news. It is arguably harder to implement the *Function over Form* strategy in a way that will be competitive with the other systems. On the other hand, a *Function over Form*-oriented system should be able to classify new names that do not occur in the gazetteers if the context surrounding the name provides reliable clues for disambiguation. Consequently, such a system should be able to keep up its classification rate on future news material without requiring continuous updates of its gazetteers.

It should also be noted that the Norwegian memory-based system achieves an F-score of 90.7 if it is tested with leave-one-out testing (cf. section 4.9), but as I argue in section 4.9, I do not believe that leave-one-out testing is the most appropriate method for this task.

The *Form over Function* strategy used by the Danish and Swedish systems seems to be closest to the one adopted for the Message Understanding Conferences (MUCs), and by extension the CoNLL competitions (Tjong Kim Sang, 2002a; Tjong Kim Sang and De Meulder, 2003), which largely follow the guidelines developed for MUC. For example, Chinchor, Brown, Ferro, and Robinson (1999, p. 13) include the following among those names that are to be tagged as locations: "Metonyms, herein designated 'common' metonyms, that reference political, military, athletic, and other organizations by the name of a city, country, or other associated location." One of the examples they provide is the following (Chinchor, Brown, Ferro, and Robinson, 1999, p. 13):

(1) Germany invaded Poland in 1939.
<B_ENAMEX TYPE="LOCATION">GERMANY<E_ENAMEX> invaded
<B_ENAMEX TYPE="LOCATION">Poland<E_ENAMEX> in ...

where the XML notation below the sentence shows that both *Germany* and *Poland* are marked as locations, not as organizations, as would be required with the Norwegian NER systems.

In fact, the named entity type of *Poland* can be considered to be vague, or underspecified, in this context—the entity being invaded could be seen both as a land area (i.e., a location) and as a political entity (i.e., an organization). Thus, it could be argued that the named entity recognizer should not be forced to make a decision here, or at least that classification as a location or as an organization should both be accepted. Nevertheless, I do require the recognizer to make a decision in all cases, partly because that is the way the corpus was annotated by Nomen Nescio, and partly because it facilitates a comparison with earlier work.

4.4 Named entity categories

As in most earlier work, I have formulated the NER task as a classification task in which the aim is to categorize a named entity into one of a number of pre-defined categories. Since the NER work presented here was done as part of the Nomen Nescio network, it uses the set of named entity categories defined by that network. The categories are as follows (see Jónsdóttir (2003) for further discussion of the guidelines used by the annotators):

- PERSON (people, animals, mythical characters, etc.)
- ORGANIZATION (companies, institutions, associations, etc.)
- LOCATION (countries, cities, mountains, etc.)
- WORK (books, movies, newspapers, etc.)
- EVENT (cultural events, sports events, etc.)
- OTHER (names that do not fit into any of the other categories)

This set extends the set of three categories (PERSON, ORGANIZATION, and LOCATION) that are found in the definition of the named entity task for the two latest Message Understanding Conferences, MUC-6 (Grishman and Sundheim, 1996) and MUC-7 (Chinchor, 1997), which has been used by most previous NER work.

4.5 Information sources for named entity recognition

I have experimented with the use of different types of information, ranging from information that can be extracted from text that has only been tokenized, to information obtained from various other sources, such as gazetteers (name lists) and the syntactic category of words in the sentence. Using text that has only been tokenized (i.e., not tagged or annotated with information of any kind) has the obvious advantage that the recognizer depends only on a minimum of existing NLP tools[2]. On the other hand, we would expect the use of additional information sources to increase the performance of the system.

As pointed out by McDonald (1996), disambiguation of named entity categories can make use of both internal and external evidence. Internal evidence consists of information pertaining only to the proper name itself, such as whether or not it is

[2]However, tokenization may benefit from other tools such as part-of-speech taggers and syntactic parsers. For example, the Oslo-Bergen tagger uses its own syntactic analysis to support tokenization of proper names.

capitalized, the identification of certain prefixes or suffixes in the name, and whether the name occurs in gazetteers. External evidence is information that is obtained from the linguistic context of the word, e.g., the identity of word forms occurring in the same sentence as the proper name, or prior classification of other occurrences of the same name within the document. In my experiments, I have used information sources that provide both internal and external evidence; the full set of information features that has been tested is discussed in section 4.8.

4.6 Training and test corpora

For training and testing of the named entity classifier, I use a part of the Oslo Corpus of Tagged Norwegian Texts that has been manually annotated with named entity categories, as was shown in chapter 3 and repeated in Figure 4.1 on page 94. In this subcorpus, markers for named entity categories are appended to the word readings, and take the following form:

- PERSON: &pe*
- ORGANIZATION: &or*
- LOCATION: &st* (from Norwegian *sted*)
- WORK: &ve* (from Norwegian *verk*)
- EVENT: &he* (from Norwegian *hendelse*)
- OTHER: &an* (from Norwegian *annet*)

This is the same corpus that is used by the other two Norwegian systems in the Nomen Nescio network, i.e., the system based on Constraint Grammar rules (Jónsdóttir, 2003) and the one that disambiguates named entities using maximum entropy modelling (Haaland, 2008). The manual annotation of the corpus was done as part of the work carried out by the Nomen Nescio network[3].

The corpus contains 226,984 tokens, of which 7275 are named entities. As shown in Table 4.2, almost half of the material in this subcorpus comes from newspapers, with the remaining half about equally divided between magazine articles and fiction material (i.e., novels).

Note that there are clear differences in the relative frequencies of named entities in these text types: While the newspaper texts constitute less than half of the total number of words, its share of proper names is almost sixty per cent. The magazine texts, on the other hand, contain about a quarter of the tokens, and this is also the

[3]The annotation work was done by Åsne Haaland, Andra Björk Jónsdóttir, and Lilja Øverlid.

	Tokens	Token %	Proper names	Proper name %
Newspapers	105,493	46.5	4545	59.9
Magazines	62,870	27.7	1926	25.4
Fiction	58,621	25.8	1119	14.7
Total	226,984	100.0	7590	100.0

Table 4.2: Distribution of tokens and proper names in the three text types that make up the subcorpus for the named entity recognition experiments: newspaper articles, magazine articles, and novels.

proportion of proper names from this genre. Finally, the number of words from fiction constitutes about a quarter of the total number of words, but its proper names make up less than fifteen per cent of all the names in the corpus.

	PERSON	LOCATION	ORG.	OTHER	WORK	EVENT	Total
Tokens	3588	1949	1295	267	137	39	7275
Proportions	49.3%	26.8%	17.8%	3.7%	1.9%	0.5%	100.0%

Table 4.3: The distribution of named entity categories in terms of tokens and proportions in the subcorpus that was used for named entity recognition.

Table 4.3 provides some information about the distribution of name categories in the corpus. The categories are ordered according to the number of names they contain. As the table shows, the proportion of names from the different categories varies considerably, from 49.3% person names to 0.5% event names. This is a result of the fact that we did not try to create a corpus in which each category was represented in equal terms; rather, we wanted a corpus that reflects the distribution of name types in authentic texts.

	PERSON	LOCATION	ORGANIZATION	OTHER	WORK	EVENT
Newspapers	40.6%	25.5%	29.1%	2.0%	2.0%	0.8%
Magazines	51.0%	28.9%	9.9%	7.6%	2.5%	0.2%
Fiction	76.8%	17.8%	2.4%	2.3%	0.6%	0.09%

Table 4.4: The proportions of different named entity categories in the various text types found in the named entity recognition subcorpus.

Finally, Table 4.4 shows the proportions of named entity categories in the different text types in the NER subcorpus. The overall prevalence of person names that was indicated in Table 4.3 is reflected in all text types. Not surprisingly, however, the newspaper texts contain a much larger proportion of organization names than the other text types do. The novels, on the other hand, contain very few organization names and a vast majority of person names. Finally, as expected, there are more references to work names (movies, books, etc.) in the newspapers and magazines than in the fiction material.

4.7 Overview of the system

4.7.1 The Oslo-Bergen tagger

The first component of the hybrid NER system presented here consists of the Oslo-Bergen tagger (Hagen et al., 2000a), described in chapter 3, which was extended with a named entity recognition component for use in the Nomen Nescio project.

The tagger performs tokenization, part-of-speech tagging, and syntactic dependency parsing before it carries out the first part of the NER task, viz. identification of word sequences that constitute proper names. This means that it can use morphological and syntactic information when identifying proper names. Access to such information is more important in Norwegian than in English, due to differences in capitalization of proper name phrases in these two languages. In most cases, Norwegian proper name phrases other than person names only have their first word capitalized (e.g., *Norges jeger- og fiskerforbund* "The Norwegian Association of Hunters and Fishermen"). This means that NP chunking is necessary in order to delimit the sequence constituting the proper name, as illustrated in (2), where the proper name is given in bold. In the original text, there is no visual indication that the proper name includes *fiskerforbund* but not *står*, so some kind of analysis is required. In the Oslo-Bergen tagger, this is done by a set of heuristic rules.

(2) **Norges jeger- og fiskerforbund** står likevel på sitt.
Norway's hunter- and fisherman-association stands still on its
"The Norwegian Association of Hunters and Fishermen still stand their ground."

Correct chunking might also depend on a deeper syntactic analysis (Hagen, 2003; Johannessen et al., 2003), for example in cases like (3)[4]:

(3) [I går ADV] [ga FV] [Hans SUBJ] [Petter I-OBJ] [en bok D-OBJ].
[Yesterday] [gave] [Hans] [Petter] [a book]
"Yesterday, Hans gave Petter a book."

Norwegian is a V2 language, meaning that the finite verb always occurs as the second constituent in declarative main clauses. Normally, in such clauses the SUBJECT is found in preverbal position, but if that position is already occupied by a another constituent (such as the topicalized ADVERBIAL in (3)), the SUBJECT is instead located directly after the verb.

In this example, both the SUBJECT *Hans* and the INDIRECT OBJECT *Petter* are male first names. Furthermore, *Hans Petter* is a perfectly fine first name, and double names of this format are very popular. Hence, if the context is disregarded, *Hans* and *Petter* could be taken to constitute a single complex person name. In order to

[4] In this example, I use the following abbreviations for syntactic categories: ADV: ADVERBIAL; FV: FINITE VERB; SUBJ: SUBJECT; I-OBJ: INDIRECT OBJECT; D-OBJ: DIRECT OBJECT.

avoid merging the SUBJECT and the INDIRECT OBJECT into one constituent, we need to perform a syntactic analysis that takes into account the ditransitive valency of the verb *ga* "gave". Since such an analysis will tell us that the clause requires an INDIRECT OBJECT, and the only possible candidate for that function is *Petter*, we can conclude that *Hans* and *Petter* must indeed be separate constituents.

Because the NER extensions to the Oslo-Bergen tagger were developed by other participants in the Nomen Nescio project, I will not go into further details concerning these extensions. The rest of this chapter will focus instead on my own contribution to the named entity recognizer, i.e., disambiguation of named entity categories for those names that are found by the tagger. For more information about the use of the Oslo-Bergen tagger for identifying (and classifying) proper names, see Jónsdóttir (2003). Hagen (2003) and Johannessen et al. (2003) provide more details about the use of chunking and syntactic analysis for identifying names.

4.7.2 Disambiguation using data-driven methods

I carry out the disambiguation subtask using data-driven methods, with a main focus on memory-based learning. For the MBL experiments described in this chapter, I use TiMBL with an IB1 database, the default database type that works in the way described in section 2.2. In the initial experiments, where parameter settings are chosen manually, I explore the effects of using information gain, gain ratio, chi-squared, and shared variance for feature weighting. Modified Value Distance Metric (MVDM; cf. section 2.2.3) is used to obtain estimates of the match between feature values. With MVDM, it can be beneficial to use values of k higher than one, meaning that not only the very closest neighbours are taken into account. Hence, I test the system with various values of k.

In the TiMBL parameter optimization experiments reported in section 4.9.5, the Overlap and Jeffrey divergence similarity metrics and various forms of distance weighting are also tested. The parameter optimization experiments that involve maximum entropy modelling test the General Iterative Scaling method (GIS; Darroch and Ratcliff (1972)) and a limited memory hill-climbing method (L-BFGS; Benson and Moré (2001); Malouf (2002a)). The L-BFGS method uses a Gaussian prior, and different values for this prior are also tested. The number of iterations is set to 100, and the remaining parameters are kept at their default values.

4.7.3 Document-centred post-processing

After having run a text through the Oslo-Bergen tagger and the data-driven disambiguator, I test the effect of what has been referred to as a "document-centred

approach" (Mikheev, 2000, 2002) or the concept of "one sense per discourse" (Gale, Church, and Yarowsky, 1992a,b; Yarowsky, 1995).

Both of these notions refer to the phenomenon that ambiguous words have a strong tendency to occur with only one of their senses within one and the same discourse or document. In this thesis, I will use the term *document-centred approach*, or *DCA*, to refer to processing that takes into account which category is the dominant one for a given name in a document and uses this information to decide on the category of other occurrences of the name that have not been disambiguated, or that have been assigned a different category at one of the earlier processing steps.

The data-driven disambiguators are forced to disambiguate all names, but their confidence in their decisions will vary. Hence, I will argue that a measure of confidence should be taken into account when deciding whether to use DCA to override the classification made by the disambiguator. In other words, some contexts provide better grounds for classification than others, making the disambiguator vary in its confidence with regard to the decisions that it makes, and highly confident decisions should be allowed to override classifications made in cases where the disambiguator is less confident.

The DCA procedure employed in the current system is as follows, given a particular name NAME and a particular document DOC:

- For each occurrence of NAME in DOC, register the confidence value that the disambiguator assigns to each category—both the selected category and the remaining ones
- When the entire document DOC has been processed, the dominant category DOM for NAME *in this particular document* is defined as the one with the highest cumulative confidence value (i.e., the highest sum over all occurrences of the name in this document)
- For each occurrence of NAME in DOC,
  - if NAME was given the category DOM by the disambiguator, do not make any changes
  - else (i.e., if NAME was given a different category by the disambiguator)
    - if the category was selected with a confidence value exceeding a certain threshold, do not make any changes
    - else change the category to DOM

Note that for a probabilistic mechanism, such as a MaxEnt model, the confidence value of a category can be taken to be the probability of selecting the category for a given name occurrence. For a non-probabilistic model such as a memory-based learner,

on the other hand, the confidence values do not represent probabilities. Rather, they represent a weighted support set (i.e., a set of nearest neighbours; Daelemans et al. (1999)) in which the absolute magnitude of each value depends on the particular instance base used. For the present purposes, however, the instance base is held constant, which makes it possible to make DCA decisions based on the relative magnitude between the confidence values given this instance base.

4.8 Experiments

I have run experiments with a varying set of features, in order to investigate the effect of different types of information on the performance of the classifier. The following types of features have been tested:

- The inflected form of the named entity. This is simply the string of word tokens that make up the name that the mechanism is currently trying to classify.

- The lemma of the named entity. The lemma is obtained from a morphological analyzer which is part of the Oslo-Bergen tagger. For known words, the lemma can be found in a lexicon, while lemmas for novel words are obtained using a compound analyzer, or "guesser" (Johannessen and Hauglin, 1998). All lemmas are decapitalized to avoid mismatches due to different capitalizations of the same name in different contexts.

- Inflected forms in the immediate context of the named entity. This is actually a collection of features, one for each position in the context that is taken into account. The size of this context window can be varied; in the experiments reported here, a window size of ± 2 words has been used.

- Lemmas in the immediate context of the named entity. This is analogous to the previous feature type, using lemmas instead of inflected forms.

- Whether the named entity consists of all capital letters. The intuition behind this is that such names are likely to be acronyms, which mostly belong to the ORGANIZATION category (e.g., IBM, EU).

- Whether all the words in the name are capitalized. As mentioned in section 4.7.1, Norwegian proper name phrases should have only their first word capitalized, unless the phrase constitutes the name of a person. Hence, information about whether all words are capitalized might help distinguish between person names on the one hand and names of, for instance, organizations and works on the other hand.

- Whether the named entity (or parts of it) occurs in one or more gazetteers. A number of gazetteers have been created by the Nomen Nescio network. These are lists of names that are known to belong to one of our categories, i.e., lists of common person names, organizations, locations, etc.

- The number of words in the name.

- Component lemmas. When a name consists of several words, there will be one feature for the lemma of each of the component words.

- Name "suffix". This is not necessarily a suffix in the morphological sense; rather, it is simply taken to be the last four characters of the (last word of the) name.

- Syntactic relations between the named entity and other parts of the sentence. The Oslo-Bergen tagger performs shallow dependency parsing. As a result of this analysis, each word in a sentence is marked either as having a syntactic category such as SUBJECT or FINITE VERB, or as a modifier or complement such as ADJ> (adjective modifying a word to its right) or <PP-UTFYLL (the complement of a preposition to the left)[5]. However, the tagger does not explicitly state which words participate in any given dependency relation. To compensate for this, I use a slightly modified version of the SPARTAN system (Velldal, 2003) to extract selected relations. These relations are as follows: SUBJECT—VERB, OBJECT—VERB, and PREPOSITION—COMPLEMENT, where the proper name functions as SUBJECT, OBJECT, and COMPLEMENT, respectively.

- The part-of-speech of the word to the left of the name or the two words to the left of the name. The parts-of-speech are taken from the output of the Oslo-Bergen tagger.

These features are coded in feature vectors of the kind that was described in section 2.2.1 and exemplified in Table 2.1. Table 4.5 provides a more realistic vector example, in which all features used in the full version of the TiMBL-based disambiguator are included. The table displays the representation of the organization name *Tine Finnmark* in the sentence *Tror dere Tine Finnmark skal klare seg som frittstående og selvstendig selskap?* "Do you think Tine Finnmark will make it as a separate and independent company?".

Note that this example poses a particular challenge for the system, since the name *Tine Finnmark* is found in the PERSON and LOCATION gazetteers, but not in the ORGANIZATION gazetteer, although the name should actually be classified as an organization. The reason for its presence in the other gazetteers is that *parts* of the name

[5] A full list of the syntactic functions used by the tagger is given at `http://www.hf.uio.no/tekstlab/tagger2.html#syntaggsett` (in Norwegian).

Feature name	Feature value
Lemma two positions to the left	*tro*
Lemma one position to the left	*dere*
Lemma of the name itself	*tine finnmark*
Lemma one position to the right	*skulle*
Lemma two positions to the right	*klare*
Only capital letters in name?	No
Only capitalized words in name?	Yes
Number of words in name	2
Part-of-speech of first word to the left	Pronoun
Part-of-speech of second word to the left	Verb
Last four letters in name ("suffix")	*mark*
Lemma no. 1 in name	*tine*
Lemma no. 2 in name	*finnmark*
Lemma no. 3 in name	
Lemma no. 4 in name	
Lemma no. 5 in name	
Lemma no. 6 in name	
Lemma no. 7 in name	
Syntactic function	SUBJECT
Found in PERSON gazetteer	Yes
Found in LOCATION gazetteer	Yes
Found in ORGANIZATION gazetteer	No
Found in WORK gazetteer	No
Found in EVENT gazetteer	No
Found in OTHER gazetteer	No
Correct category	ORGANIZATION

Table 4.5: The TiMBL feature vector representation of the organization name *Tine Finnmark* in the context *Tror dere Tine Finnmark skal klare seg som frittstående og selvstendig selskap?* "Do you think Tine Finnmark will make it as a separate and independent company?"

are found in those gazetteers: the first part, *Tine*, is a woman's name, while *Finnmark* is the name of the northernmost area of Norway. When a match is found for part of the name, the whole name is marked as a match as well.

For instance, if we want to classify the man's name *Harald Milli*, the first name *Harald* can be found in our PERSON gazetteers, and hence the whole name *Harald Milli* can be correctly marked as a person name. Another example is the phrase *Guatemala by* "Guatemala City", which will be marked as a possible location because the *Guatemala* part is found in our LOCATION gazetteers. Hence, the technique of "propagating" gazetteer categories of subparts up to the whole name may provide important information to the classification process. It is therefore considered valuable, although it may occasionally be detrimental, such as in the example in Table 4.5, where

an organization name is constructed from parts that look like a person name and a location name.

4.9 Results and evaluation

The memory-based classifier has been trained and tested in two ways. Firstly, I have used TiMBL's capacity to do leave-one-out testing (LOO) (Weiss and Kulikowski, 1991), since it allows me to test each name in the corpus against all the others, thereby maximizing the size of both training and test material. Secondly, I have applied the more common method of 10-fold cross-validation (10CV) (Weiss and Kulikowski, 1991), because it makes it possible to compare my results to results from experiments on the same corpus using other methods for which leave-one-out testing is computationally unfeasible (cf. section 2.5.3).

These two testing methodologies have been applied in two batches, the first one using TiMBL's default parameter settings and the second one with a limited form of parameter optimization. This optimization procedure consists of manually selecting a few values for each input parameter of the machine learning algorithm and testing all combinations of these values. Finally, I have used the Paramsearch program (van den Bosch, 2004) to do a much more extensive search for optimal parameter values, before re-running the 10CV experiments using these values. This allows me to examine the value of automatic parameter optimization over the amount of optimization that can feasibly be done by hand.

4.9.1 Default settings

The first batch of experiments has been performed with TiMBL using its default parameter settings. The default values of the relevant settings (i.e., all the settings that will be optimized later in this chapter) are shown in Table 4.6. The metric threshold is not used when the default Overlap metric is selected. When the distance metric employed is MVDM or Jeffrey divergence, however, this threshold determines the lowest frequency at which MVDM or Jeffrey is applied; for feature values below this threshold, Overlap is used instead (consequently, the default value of 1 means that MVDM/Jeffrey is always used). Like all other TiMBL experiments described in this thesis, these experiments use the IB1 algorithm, which is the default learning algorithm used by TiMBL.

Distance metric	Metric threshold	Weighting scheme	k	Extrapol. method
Overlap	1 (not used)	Gain ratio	1	None

Table 4.6: TiMBL's default parameter settings.

In the cross-validation experiments, the system is trained on approximately 90% of the corpus and then tested on the remaining 10%. This is repeated 10 times (i.e., for 10 *folds*), with a different part of the corpus being used as a test corpus each time. All documents in the corpus are entirely included in either the test corpus or the training corpus. In other words, the split between training and test corpora is always located at a document boundary. As I argue below, the results from the experiments indicate that this is important in order to obtain a fair evaluation of the system. On the other hand, it means that there is a certain variation in the sizes of the training and test corpora used in the different folds. Table 4.7 shows the number of items in the training and test corpora for each fold.

Fold number	1	2	3	4	5	6	7	8	9	10
Training items	6725	6896	6874	6692	6799	6565	6714	6688	6943	6937
Test items	812	641	663	845	738	972	823	849	594	600

Table 4.7: The number of training and test items in each fold of the 10-fold cross-validation experiments. In each fold, the number of training and test items add up to 7537.

Features	LOO accuracy	10CV accuracy/std.dev.
IF	77.66	68.07 ±3.80
L	77.80	68.53 ±3.69
L+AC	78.53	69.40 ±3.80
L+GZ	83.49	76.37 ±4.16
L+NW	79.14	68.50 ±3.79
L+OC	79.29	68.86 ±3.72
L+CL	87.53	73.64 ±4.32
L+SUF	86.48	71.88 ±3.84
L+SYN	74.72	66.54 ±3.93
L+POS	75.67	67.44 ±3.29
L+POS2	75.02	66.51 ±3.24
L+ALL	88.62	78.98 ±3.23
L+ALL+DCA	87.94	79.12 ±4.77

Table 4.8: Results of leave-one-out testing (LOO) and 10-fold cross-validation (10CV) using default parameter settings for TiMBL. IF = inflected forms; L = lemmas; AC = all capital letters; GZ = gazetteers; NW = number of words; OC = all words capitalized; CL = component lemmas; SUF = "suffixes" (four last characters of the name); SYN = syntactic information; POS = part-of-speech of word to the left; POS2 = part-of-speech of two words to the left; ALL = all features; DCA = document-centred approach.

Results from 10CV and LOO runs are shown in Table 4.8. The table reports on the classifier accuracy obtained for each feature combination. For the 10CV experiments, the accuracy is the average of the accuracy scores obtained in the 10 folds, and the standard deviation of the accuracy is also shown.

Perhaps the most striking aspect of the figures in Table 4.8 is the big difference between LOO and 10CV accuracy scores: in the LOO experiments, we obtain much higher accuracy figures in all conditions than we do in the 10CV experiments. The optimal classifier is the one that includes all features as well as document-centred post-processing, and for this classifier we obtain accuracy scores of 87.94% and 79.12% for LOO and 10CV testing, respectively.

An advantage of using LOO is that it allows us to use the entire corpus for both training and testing, and an increased training corpus is expected to yield better results. This expectation is supported by the figures in Table 4.9, which shows the number of names along with recall, precision, and $F_{\beta=1}$ score for each category when used with 10CV and all features plus DCA. The F-scores correlate strongly with the proportion of names from each category, which indicates that 10CV results might improve with a larger training corpus.

	PERSON	LOCATION	ORGANIZATION
Number of items	3677	1912	1501
Recall	92.60 ±3.97	79.92 ± 5.45	63.61 ±7.13
Precision	85.84 ±6.39	73.25 ±10.78	72.11 ±5.37
$F_{\beta=1}$	89.02 ±4.76	76.02 ± 7.71	67.42 ±5.32
	WORK	OTHER	EVENT
Number of items	145	263	39
Recall	24.88 ±14.95	14.73 ±15.25	5.00 ±11.25
Precision	43.26 ±20.31	33.13 ±25.77	15.00 ±33.75
$F_{\beta=1}$	29.00 ±12.83	18.75 ±15.41	7.50 ±16.87

Table 4.9: Number of names along with cross-validation recall, precision and F-score for each name category.

However, there is also another, more likely source of the big difference between the two result sets. The use of LOO testing means that when the system classifies some name, any other occurrence of the same name within the same document will be present in the database. With 10CV, on the other hand, I split the training and test sets on document boundaries (in order to be able to apply document-centred post-processing), so that all occurrences of a name in a certain document are found either in the training set or in the test set; there is no risk that some of them occur in one corpus and some in the other.

Thus, using LOO testing for this kind of task may be considered "cheating" in a certain sense. We may reasonably assume that particular proper names tend to occur only in certain documents, and that they tend to belong to a certain category in those documents (that is, after all, the foundation for the document-centred approach). When testing a certain name using LOO, then, the training set will often contain a

number of occurrences of this particular name, so that the classifier will find training instances that exactly match the inflected form or lemma of the name. Furthermore, these training instances will most likely be assigned to the category that is the correct one for this name in this particular document.

With 10CV, on the other hand, there is a good chance that the training set will not contain any occurrences of this particular name, making the task harder for the classifier because it cannot find any exact matches in the instance base. Furthermore, even if there are occurrences of the same name, they will come from different documents than the one that the testing instance is taken from, and in those other documents the name may have been assigned to different categories.

Thus, we may conclude that LOO is not a proper testing methodology for the NER task. More generally, we may question the validity of LOO testing for tasks where the presence of instances from the same document may exert an improper influence on the ability of the system to make a correct classification.

4.9.2 Manual parameter optimization

In the manual parameter optimization experiments, both LOO and 10CV have been performed with the Modified Value Difference Metric (MVDM; see section 2.2.3) and with all of the four weighting schemes offered by TiMBL (information gain, gain ratio, chi-squared, and shared variance), combined with a selection of k-values for the k-nearest neighbour algorithm. When MVDM is used, it is usually beneficial to use k-values higher than one, and odd values tend to work better than even ones (Daelemans et al., 2004). Consequently, I have tested k-values of 5, 11, 19, and 25.

Results from 10CV and LOO runs are shown in Table 4.10. The table reports on the highest classifier accuracy that is reached for each feature combination, along with the k-value (among the tested values of 5, 11, 19, and 25) which yields this accuracy.

The choice of feature-weighting scheme is not listed in the table, because it turns out that gain ratio gives the best results in almost all cases. The only exception is the IF case with LOO, where chi-squared performs better than gain ratio, though not significantly better.

Note that since the figures in Table 4.10 are the result of picking the parameter values that lead to the optimal performance for each classifier and with each testing methodology, they cannot be said to result from a proper evaluation of the systems. In order to get a proper evaluation, we need to run the classifiers on test data which were not included in the material on which the parameters were optimized. A proper evaluation will be presented in section 4.9.5.

All experiments include as features the inflected forms or lemmas occurring in a context window centred on the name, and these features will have many low-frequent values. According to Daelemans et al. (2004), the chi-squared statistic tends to pro-

Features	LOO k-value	LOO accuracy	10CV k-value	10CV acc./std.dev.
IF	19	85.39	25	72.50 ±4.25
L	19	86.64	25	73.71 ±4.11
L+AC	19	86.60	25	73.96 ±3.99
L+GZ	11	88.40	25	79.11 ±3.85
L+NW	25	87.05	25	74.04 ±4.01
L+OC	25	87.13	25	74.40 ±3.92
L+CL	19	88.74	25	77.09 ±4.01
L+SUF	11	87.41	19	72.66 ±3.65
L+SYN	25	86.80	25	74.36 ±3.84
L+POS	19	86.92	25	74.28 ±4.27
L+POS2	19	86.73	25	73.96 ±4.27
L+ALL	5	90.66	5	81.89 ±3.28
L+ALL+DCA	5	90.67	5	83.16 ±3.09

Table 4.10: Results of leave-one-out testing (LOO) and 10-fold cross-validation (10CV), along with optimal choices of k for the k-nearest neighbour classifier determined by manual optimization. IF = inflected forms; L = lemmas; AC = all capital letters; GZ = gazetteers; NW = number of words; OC = all words capitalized; CL = component lemmas; SUF = "suffixes" (four last characters of the name); SYN = syntactic information; POS = part-of-speech of word to the left; POS2 = part-of-speech of two words to the left; ALL = all features; DCA = document-centred approach.

duce poor results in such a situation, and this has been confirmed by my experimental results. When chi-squared is corrected for degrees of freedom, yielding a shared variance measure (Daelemans et al., 2004), the results are somewhat better. However, in general, the use of gain ratio produces the best results.

4.9.3 Effects of different k-values

Another noticeable difference between the two training and testing schemes is that with 10CV the system performs best with the largest k-value in all but one of the cases where only selected features are used, while with LOO there is considerably more variation in the choice of optimal k-value. Again, this could be due to the fact that with LOO the instance base will contain more instances that are similar or identical to the test instance, meaning that the best support instances will often be found in a narrower neighbourhood.

With both schemes, the smallest k-value turns out to be optimal when all features are included, while the models with only selected features perform best when higher k-values are used. A plausible explanation for this is that a larger amount of information about the instances is likely to bring out their similarities and differences more clearly. Thus, with more features, similar instances will tend to be more tightly clustered in the instance space, and hence a small k-value, which restricts the search for matches to a very narrow neighbourhood, will provide the most reliable evidence for classification.

It has been verified, however, that k-values smaller than 5 do not lead to further improvement.

4.9.4 Effects of individual features

This section describes the positive or negative effects of the various features. To test for statistical significance, I have applied McNemar's test to pairs of classifiers. Unless stated otherwise, the reported differences are significant at the 1% level.

In the first row of Table 4.10 (IF), the only features used are the inflected forms of the proper name and the words in its immediate context. Obtaining values for these features is very simple in that it only requires tokenization of the text, and the result of this experiment can be viewed as a baseline for the performance of the MBL-based classifier. This classifier obtains an LOO accuracy of 85.39% and a 10CV accuracy of 72.50%.

Exchanging inflected forms for lemmas (L) yields a significant performance increase (LOO: 86.64%; 10CV: 73.71%). This is to be expected, as the use of lemmas abstracts away from inflectional information, which is unlikely to be important for this task, and at the same time reduces the sparseness of the data. The rest of the feature combinations all involve the use of lemmas instead of inflected forms, and further tests for statistical significance are made against the L case.

The two feature types giving the largest performance increase are component lemmas (L+CL; LOO: 88.74%; 10CV: 77.09%) and gazetteers (L+GZ; LOO: 88.40%; 10CV: 79.11%). A topic for further research might be to investigate whether gazetteers containing only a limited selection of names would yield comparative results, as suggested by Mikheev et al. (1999).

Another feature leading to a smaller but still significant increase is information about whether all words of a multi-word name are capitalized (L+OC; LOO: 87.13%; 10CV: 74.40%). The number of words in the name (L+NW) is significant with LOO (87.05%) but not with 10CV (74.04%). The four-letter suffix of the name (L+SUF) boosts performance significantly with LOO (87.41%) but actually lowers it with 10CV (72.66%).

The feature type that contributes the highest level of linguistic information is the one that involves syntactic relations between names and verbs or prepositions (L+SYN). Although this feature type improves the performance of the system significantly with 10CV (74.36%), with LOO (86.80%) it does not. In order to check whether this could be due to data sparseness, I have run additional experiments where I only include relations which occur at least three times in the corpus. Also, since this feature type only applies to names that participate in one of the selected relations, I have compared the performance levels of the L and L+SYN classifiers on these names

only. However, none of these conditions produce a significant performance increase with LOO.

Other features that use the result of linguistic processing are the part-of-speech features. L+POS gives a significant increase at the 5% level (LOO: 86.92%; 10CV: 74.28%), while L+POS2 does not (LOO: 86.73%; 10CV: 73.96%). The part-of-speech of the second word to the left receives a very low gain ratio weight value, which is a further indication that this is not a good feature for the present task.

Finally, the L and L+AC cases do not differ significantly (LOO: 86.60%; 10CV: 73.96%). I suspect that this could be due to the use of newspaper texts which contain many person and location names with all capital letters (i.e., names of journalists and their locations occurring at the beginning of articles).

The document-centred post-processing step boosts performance significantly with 10CV: the optimal classifier, which includes all features, obtains accuracy scores of 83.16% with DCA and 81.89% without. Using LOO testing, on the other hand, there is virtually no increase. This result supports the hypothesis presented earlier, viz. that LOO makes good use of occurrences of the same name within the same document. With LOO testing, the instance base contains all occurrences of a name in a document except the one being tested, and this will make the classifier itself perform much the same process as DCA does. Thus, this lack of performance increase with DCA and LOO should be expected.

The results given in Table 4.10 are the overall accuracy measures for all names in the corpus. However, as was shown in Table 4.9, there is considerable variation in the performance levels reached for the different categories. Most strikingly, the system exhibits very high performance on the PERSON category compared to the other categories, and since about half of all names in the corpus are person names (cf. Table 4.3), this explains the high overall performance of the system.

Thus, in spite of the high overall performance, it is important to note that the system does not perform equally well on all categories. On the other hand, it is also worth noting that the system performs best on those categories (PERSON, ORGANIZATION, and LOCATION) that were used in the Message Understanding Conferences (Chinchor, 1997; Grishman and Sundheim, 1996) as well as in the CoNNL-2002 and CoNNL-2003 shared tasks of language-independent named entity recognition (which also included a MISCELLANEOUS category) (Tjong Kim Sang, 2002a; Tjong Kim Sang and De Meulder, 2003).

4.9.5 Automatic parameter optimization using Paramsearch

Although I tested a variety of parameter settings in the LOO and 10CV experiments reported so far, it was practically infeasible to test all possible parameter setting combinations by hand. Thus, although I tested all four weighting schemes offered by

Features	TiMBL Paramsearch accuracy/std.dev.
IF	84.00 ±1.41
L	82.51 ±3.97
L+AC	83.32 ±3.13
L+GZ	83.66 ±3.70
L+NW	84.92 ±1.00
L+OC	82.94 ±4.14
L+CL	86.91 ±1.78
L+SUF	82.81 ±4.66
L+SYN	82.78 ±3.60
L+POS	85.19 ±1.38
L+POS2	85.33 ±1.32
L+ALL	87.26 ±3.21

Table 4.11: TiMBL accuracy figures reported by Paramsearch averaged over 10 folds. Note that these figures are obtained by optimizing classifier performance on the test data, and hence do not represent a proper evaluation of the system. See Table 4.12 for the results of a proper cross-validation evaluation. IF = inflected forms; L = lemmas; AC = all capital letters; GZ = gazetteers; NW = number of words; OC = all words capitalized; CL = component lemmas; SUF = "suffixes" (the last four characters of the name); SYN = syntactic information; POS = part-of-speech of word to the left; POS2 = part-of-speech of two words to the left; ALL = all features; DCA = document-centred approach.

TiMBL, I only tested a few k-values. Furthermore, the experiments were restricted to using the MVDM similarity metric, and the cutoff frequency at which TiMBL changes from MVDM to the Overlap metric was held at its default value of 1 (meaning that MVDM is always used instead of Overlap).

However, an automatic and more exhaustive search for optimal parameters can be carried out using the Paramsearch program (van den Bosch (2004); cf. section 3.7). At the end of optimization, Paramsearch reports on the performance score obtained on the test data in the final iteration. These performance scores are displayed in Table 4.11. Note that, since the DCA post-processing is external to TiMBL, it is not included in the parameter optimization process and hence not reported in the table.

A comparison with the figures in Table 4.10 on page 82, which resulted from manual parameter optimization, shows that there are huge improvements to be gained from running automatic optimization. Focusing on the optimal classifier without DCA, which was listed in the next-to-last row of Table 4.10, we see an increase in accuracy from 81.89% to 87.26% when changing from manual to automatic optimization. This large performance boost agrees with the finding by van den Bosch (2004) that TiMBL (using the IB1 algorithm) benefits greatly from pseudo-exhaustive parameter optimization.

There is one caveat, however: as the above discussion of the leave-one-out results shows, NER is an example of a task in which training and test data should not be

drawn from the same document, because a particular name will typically be classified in a single way within one and the same document, and thus its appearance in the training data will give the algorithm an unfair advantage when classifying this name. Paramsearch creates its own split into training and test data, and the user has no way of specifying that the split should always happen on a document boundary. For this reason, the difference between the results from manual and automatic optimization might be somewhat exaggerated, although the effect should be nowhere near the effect of doing leave-one-out testing.

In the experiments reported so far, the parameters have been selected by optimizing the performance on test data. Thus, as in the manual optimization experiments, the performance scores do not reflect the ability of the classifiers to generalize to new data; in other words, the so-called test data function as development data rather than as proper test data.

Thus, in order to carry out a proper evaluation of the classifiers, I adopt the strategy employed by van den Bosch (2004) and carry out 10-fold cross-validation experiments with parameter optimization. In each fold, the data is divided into the usual 90/10 proportions, and Paramsearch is used to optimize the parameter settings on the 90% part. After optimization, the classifier is tested on the 10% part that was not included in the optimization process. As in the cross-validation experiments reported earlier, the division between the optimization data and the held-out test data is always made on a document boundary. Hence, in these experiments we are guaranteed that the optimization data and the test data do not contain any material from the same documents, yielding a proper evaluation of the classifiers. The performance scores obtained in this way are shown in Table 4.12. The best classifier reaches a cross-validation accuracy of 82.48%, which is significantly better than the 79.12% obtained using default parameters ($p \ll 0.01$).

Arriving at a single set of parameter settings

Although the procedure described above constitutes a proper evaluation of the classifiers, it does not in itself provide us with a single set of parameter settings that can be used in future applications of the classifiers. Moreover, since a different set of parameter values is used in each fold, the procedure does not tell us how well each classifier would perform if it was forced to employ one particular set of parameter values on all test data—in other words, how well we can expect it to perform on future application data.

In order to solve these shortcomings, I use the following approach: I construct a single set of parameter values for each feature combination by selecting, for each parameter, the value that is found to be optimal in the majority of folds in the 10-fold cross-validation experiment described above. In the case of a tie, the value is selected

Features	TiMBL optimized 10-fold CV accuracy/std.dev.
IF	72.53 ±4.48
L	73.09 ±4.45
L+AC	73.55 ±4.46
L+GZ	78.65 ±3.98
L+NW	73.64 ±4.01
L+OC	74.02 ±3.76
L+CL	77.13 ±4.05
L+SUF	72.63 ±3.68
L+SYN	73.90 ±3.89
L+POS	73.57 ±4.00
L+POS2	73.50 ±4.11
L+ALL	81.18 ±3.25
L+ALL+DCA	82.48 ±3.03

Table 4.12: TiMBL results of 10-fold cross-validation (CV) experiments with parameter settings automatically optimized separately for each fold. Average accuracy and standard deviation over the 10 folds is shown. IF = inflected forms; L = lemmas; AC = all capital letters; GZ = gazetteers; NW = number of words; OC = all words capitalized; CL = component lemmas; SUF = "suffixes" (the last four characters of the name); SYN = syntactic information; POS = part-of-speech of word to the left; POS2 = part-of-speech of two words to the left; ALL = all features; DCA = document-centred approach.

randomly among the tying values.

The sets of parameter values constructed in this way are tested by re-running the cross-validation experiments, this time using the same set of parameter values in each fold rather than optimizing the values for each fold. The results of these experiments are shown in Table 4.13. The complete classifier with DCA reaches an accuracy of 82.53%, which is not significantly different from the 82.48% obtained by using different parameter values in each fold.

Thus, we may conclude that the results of optimizing the classifiers separately in each fold do in fact give us a fairly good indication of the performance level we obtain from extracting a single set of parameter values.

Another point to note about the results in Table 4.13 is that, as in the experiments that use manually selected parameter values, Paramsearch also finds gain ratio to be the best weighting scheme in all cases, and the default value of 1 for the threshold value for switching from MVDM or Jeffrey to the Overlap metric is always found to be optimal. Furthermore, although the optimal k-value is in most cases found to be higher than those tested in the experiments with manual parameter value selection, a value of 5 is still found to be best for the best-performing classifiers which include all features. However, for some of the cases, including the best-performing ones, the Jeffrey similarity metric, which was not tested manually, turns out to yield better results than MVDM.

Features	Distance metric	Metric threshold	Weight. scheme	k	Extrapol. method	10CV accuracy and std.dev.
IF	MVDM	1	GR	35	ID/IL	72.79 ±4.23
L	MVDM	1	GR	35	ID	73.85 ±3.95
L+AC	MVDM	1	GR	35	ED1/IL	74.28 ±4.03
L+GZ	Jeffrey	1	GR	35/11	IL	78.88 ±3.95
L+NW	MVDM	1	GR	25	ED1	74.00 ±4.11
L+OC	MVDM	1	GR	35	ED1	74.51 ±4.06
L+CL	Jeffrey	1	GR	35	ED1	77.18 ±3.82
L+SUF	MVDM	1	GR	35/15	ED1	73.02 ±3.48
L+SYN	MVDM	1	GR	25	ED1/ID	74.31 ±3.82
L+POS	Jeffrey	1	GR	25	ID	73.64 ±3.96
L+POS2	MVDM	1	GR	25	ID/Z	73.89 ±4.11
L+ALL	Jeffrey	1	GR	5	IL	80.33 ±3.09
L+ALL+DCA	Jeffrey	1	GR	5	IL	82.53 ±2.47

Table 4.13: Results of 10-fold cross-validation (10CV) experiments with TiMBL using the optimal parameter values found by Paramsearch. IF = inflected forms; L = lemmas; AC = all capital letters; GZ = gazetteers; NW = number of words; OC = all words capitalized; CL = component lemmas; SUF = "suffixes" (the last four characters of the name); SYN = syntactic information; POS = part-of-speech of word to the left; POS2 = part-of-speech of two words to the left; ALL = all features; DCA = document-centred approach.

4.9.6 Replacing MBL with MaxEnt

One of the data-driven methods that have produced very good results in earlier work is maximum entropy (MaxEnt) modelling. In MUC-7, for instance, the best performing system was a hybrid that combined hand-written rules with a MaxEnt model (Mikheev et al., 1998). Furthermore, in the CoNLL-2003 shared task, the top three systems for English and the top two results for German made use of MaxEnt modelling (Tjong Kim Sang and De Meulder, 2003). It is therefore interesting to compare the performance of MaxEnt modelling vs. memory-based learning on the classification of Norwegian named entities.

In order to compare the two modelling techniques, I have applied Zhang Le's maximum entropy modelling toolkit (cf. 3.5) to the same data as was used in the TiMBL experiments, employing Paramsearch to optimize the parameters of the MaxEnt toolkit. I have used the same set of features as those used with the memory-based learner, but without the document-centred post-processing (which is independent of the particular machine learning algorithm that is selected).

Table 4.14 on page 90 shows the accuracy figures obtained in the optimization experiments, and Table 4.15 on page 91 reports on the results of applying these optimized parameters in a 10-fold cross-validation experiment. Finally, in Table 4.16 on page 92 I show the parameter values that proved to be optimal in the majority of

folds (with tying values separated by slashes), and the accuracy figures obtained by applying this single set of parameter values in all folds.

A comparison between Table 4.15 on page 91 and Table 4.12 on page 87 shows that TiMBL consistently outperforms the MaxEnt model when parameters are optimized separately for each fold. TiMBL obtains better scores than the MaxEnt model for all feature combinations, and in the L+ALL case it reaches an accuracy of 81.18%, which is significantly better than the 80.03% obtained by the MaxEnt model ($p \leq 0.00791$). This is an interesting result considering the good performance of MaxEnt models and the modest performance of memory-based learners in earlier NER experiments reported in the literature. When a single set of optimized parameters is applied (cf. Tables 4.13 and 4.16), the difference is no longer significant, but TiMBL still performs as well as MaxEnt (TiMBL: 80.33%; MaxEnt: 80.06%; $p \leq 0.735$).

Of course, it is conceivable that the picture might have been different with a different set of features. Above all, the MaxEnt modelling technique, with its tolerance for non-independent features, might have been able to take advantage of complex features consisting of combinations of the simple features used here.

For example, most Norwegian person names contain two or three words. Furthermore, all words in a person name are capitalized. Thus, a name which contains two or three words may very well be a person name, and one which contains only capitalized words is also likely to be a person name. However, names which fulfil *both* of these criteria—i.e., two- or three-word names in which each word is capitalized—have an even higher probability of being a person name. After all, two-word names with, say, all capital letters often denote organizations (e.g. *COWI AS* "COWI Inc."), and the same is the case with, say, four-word names in which all words are capitalized (e.g. *Norsk Country Musikk Forbund* "The Association for Norwegian Country Music").

As mentioned in section 2.3, MaxEnt models have the ability to properly capture the added probability of being a person name if a feature representing the combination of the two simple features is used, because these models do not assume statistical independence between features. Thus, a topic for future research might be to create a feature set which lets the MaxEnt algorithm take advantage of its tolerance for non-independent features.

Further work on MaxEnt modelling for Norwegian NER has very recently been presented by Åsne Haaland (Haaland, 2008), who uses features that are similar to those employed here. Through more extensive experiments with MaxEnt modelling for NER (Haaland devotes her entire thesis to this field), she is able to obtain an accuracy of 81.36% for her MaxEnt model, outperforming both the MaxEnt model and the memory-based learner presented here when document-centred post-processing is not used. With the addition of DCA, however, the memory-based system described in the present chapter performs even better, given its accuracy of 82.53%.

Features	MaxEnt Paramsearch accuracy/std.dev.
IF	77.20 ±6.07
L	78.24 ±6.05
L+AC	78.00 ±7.46
L+GZ	82.53 ±2.11
L+NW	80.70 ±2.92
L+OC	80.58 ±1.94
L+CL	82.98 ±4.47
L+SUF	81.45 ±5.33
L+SYN	77.69 ±4.74
L+POS	78.10 ±6.08
L+POS2	79.85 ±3.66
L+ALL	85.93 ±3.39

Table 4.14: MaxEnt accuracy figures reported by Paramsearch averaged over 10 folds. Note that these figures are obtained by optimizing classifier performance on the test data, and hence do not represent a proper evaluation of the system. See Table 4.15 for the results of a proper cross-validation evaluation. IF = inflected forms; L = lemmas; AC = all capital letters; GZ = gazetteers; NW = number of words; OC = all words capitalized; CL = component lemmas; SUF = "suffixes" (the last four characters of the name); SYN = syntactic information; POS = part-of-speech of word to the left; POS2 = part-of-speech of two words to the left; ALL = all features.

In addition to these cross-validation results, I have applied my memory-based recognizer to a test corpus of newspaper texts containing approximately 1800 proper names. Haaland's MaxEnt-based system, as well as the other NE recognizers developed in the Nomen Nescio network, have been applied to the same corpus, and we have reported the results in Johannessen et al. (2005). On the test corpus, my memory-based system achieves an F-score of 68, while the MaxEnt classifier gets an F-score of 60. Haaland (2008) reports that a later version of her system achieves an F-score of 63 on this test corpus.

4.9.7 Varying the size of the training corpus

So far it has been shown that a memory-based named entity recognizer for Norwegian can be made to perform well (both compared to earlier results using MBL and compared to a maximum entropy modelling approach) when trained on a corpus containing approximately 7000 names. An interesting question, however, is whether we would have been able to get even better results if the training corpus had been larger. A common way of estimating this is to train the recognizer on a gradually increasing training corpus and evaluate it at each point.

Figure 4.2 on page 95 shows the results of this kind of evaluation of both the memory-based recognizer and the one based on maximum entropy modelling. I started by splitting the corpus 10 times into 90% training material and 10% test material, as

Features	MaxEnt optimized 10-fold CV accuracy/std.dev.
IF	69.82 ±6.40
L	70.48 ±5.75
L+AC	71.58 ±3.78
L+GZ	77.11 ±3.75
L+NW	72.77 ±2.73
L+OC	72.44 ±3.08
L+CL	73.37 ±3.60
L+SUF	72.32 ±4.64
L+SYN	70.70 ±3.48
L+POS	71.46 ±4.35
L+POS2	72.13 ±4.29
L+ALL	80.03 ±3.47

Table 4.15: MaxEnt results of 10-fold cross-validation (CV) experiments with parameter settings automatically optimized separately for each fold. Average accuracy and standard deviation over the 10 folds is shown. IF = inflected forms; L = lemmas; AC = all capital letters; GZ = gazetteers; NW = number of words; OC = all words capitalized; CL = component lemmas; SUF = "suffixes" (the last four characters of the name); SYN = syntactic information; POS = part-of-speech of word to the left; POS2 = part-of-speech of two words to the left; ALL = all features.

in an ordinary cross-validation experiment. I then reduced the training corpora for each fold to 3000 names and started a sequence of 10-fold cross-validation experiments in which the training corpora were gradually increased, 100 names at a time, until they reached their full size of approximately 7000 names.

In each fold of each cross-validation experiment, Paramsearch was used to perform automatic parameter optimization on the training corpus, and then the recognizer was applied to the held-out test corpus. The accuracy scores shown in Figure 4.2 represent the average of the 10 folds.

The figure clearly indicates that, both for the memory-based learner and for the MaxEnt model, a ceiling has been reached long before the corpus reaches its full size. In other words, there is no reason to believe that a larger corpus would have yielded better results for any of these methods.

4.10 Conclusions

In this chapter I have presented a named entity recognizer for Norwegian. As will be shown in chapter 8, doing named entity recognition as part of the pre-processing of the text that is fed into the anaphora resolution system leads to a marked improvement in the performance of that system. Additionally, a named entity recognizer has great value for a range of other NLP applications, such as information extraction, question answering, and machine translation systems.

Features	Parameter estimation	Gaussian prior	10CV accuracy/std.dev.
IF	L-BFGS	100	71.41 ±4.08
L	L-BFGS	60	71.84 ±3.45
L+AC	L-BFGS	0	70.37 ±4.41
L+GZ	L-BFGS	0	76.82 ±3.92
L+NW	L-BFGS	0/60	70.29 ±2.90
L+OC	L-BFGS	0/100	69.64 ±3.32
L+CL	L-BFGS	0/60/100	73.56 ±4.06
L+SUF	L-BFGS	0	69.75 ±3.52
L+SYN	L-BFGS	0	69.32 ±3.85
L+POS	L-BFGS	0	70.06 ±3.89
L+POS2	L-BFGS	5	72.71 ±4.17
L+ALL	L-BFGS	0/0.01	80.06 ±2.73

Table 4.16: Results of 10-fold cross-validation (10CV) experiments with MaxEnt using the optimal parameter values found by Paramsearch. IF = inflected forms; L = lemmas; AC = all capital letters; GZ = gazetteers; NW = number of words; OC = all words capitalized; CL = component lemmas; SUF = "suffixes" (the last four characters of the name); SYN = syntactic information; POS = part-of-speech of word to the left; POS2 = part-of-speech of two words to the left; ALL = all features.

I have investigated the contributions of different input features, and have found that most of the features I tested led to significant improvements with leave-one-out testing and/or 10-fold cross-validation, but that the most important improvements stemmed from gazetteers and the lemmas of the words that constitute the proper name phrase. This work has also led to a number of other interesting observations:

- **Memory-based learning performs as well as or better than maximum entropy modelling on this task.** Contrary to the trends found in the existing literature on named entity recognition, for the system described here, memory-based learning turns out to provide the best results. I have proposed that by narrowing the application of the machine learning component to the classification subtask, rather than to the entire combined identification and classification task that it has traditionally been applied to, the task has been recast in a way that greatly benefits the memory-based learning mechanism and enables it to perform as well as maximum entropy modelling, and even better under some conditions (separate parameter optimization for each fold).

- **Document-centred post-processing using classifier confidence improves performance.** The idea of using information from the entire document to classify individual test cases was inspired by the work of Andrei Mikheev on text normalization (Mikheev, 2000, 2002), and a certain form of DCA was used already by Mikheev et al. (1998, 1999). However, to my knowledge, using classifier confidence to aid the DCA, as described in this thesis, is a novel approach.

- **Leave-one-out testing does not seem to be appropriate for document-centred tasks.** I have showed that document-centred post-processing is a highly valuable step when it is evaluated through cross-validation, but not when leave-one-out testing is used. I have proposed that the reason for this is that the two contribute much of the same information, and hence that the contribution from DCA will not be registered when we use leave-one-out testing. In other words, the lack of a significant performance improvement shown by DCA in this case could be explained by the use of an inappropriate testing methodology. Consonant with this reasoning, I draw the more general conclusion that leave-one-out testing is not appropriate for tasks where the results are influenced by the presence (or absence), in the database, of material from the same document as the test item.

- **Automatic parameter optimization leads to huge performance improvements compared to doing limited manual optimization**. The benefits of doing extensive optimization agree with the findings of van den Bosch (2004) on various NLP tasks and of Hoste (2005) for coreference resolution.

```
Turkey      "<Tyrkia>"
                "Tyrkia" subst prop &or*
has         "<har>"
                "ha" verb pres pa1 a6 d5 rl6 tr6 d6/til pa3 tr12 <aux1/perf_part> pa6
put         "<satt>"
                "sette" verb perf-part tr1 rl4 pa1 a6 rl6 pa1/til pa2 pa5 pa3 a9 tr23
Norway      "<Norge>"
                "Norge" subst prop &or*
on          "<på>"
                "på" prep
its         "<sin>"
                "sin" det poss mask ent
red         "<røde>"
                "rød" adj pos be ent
list        "<liste>"
                "liste" subst mask appell ub ent
            "<,>"
                "$," <komma>
and         "<og>"
                "og" konj clb
Penguin     "<Penguin>"
                "Penguin" subst prop &an*
to          "<til>"
                "til" prep
Turkey      "<Tyrkia>"
                "Tyrkia" subst prop &st*
is          "<er>"
                "være" verb pres a5 pr1 pr2 <aux1/perf_part>
now         "<nå>"
                "nå" adv
a           "<et>"
                "en" det kvant noeyt ent
dead        "<dødt>"
                "død" adj pos noeyt ub ent
project     "<prosjekt>"
                "prosjekt" subst noeyt appell ub ent
            "<,>"
                "$," clb <komma>
states      "<fastslår>"
                "fastslå" verb pres tr1 tr2
Jørgensen   "<Jørgensen>"
                "Jørgensen" subst prop &pe*
            "<.>"
                "$." clb <punkt> <<<
```

Figure 4.1: Excerpt from the portion of the Oslo Corpus that has been marked with named entity tags, as discussed in section 3.2.

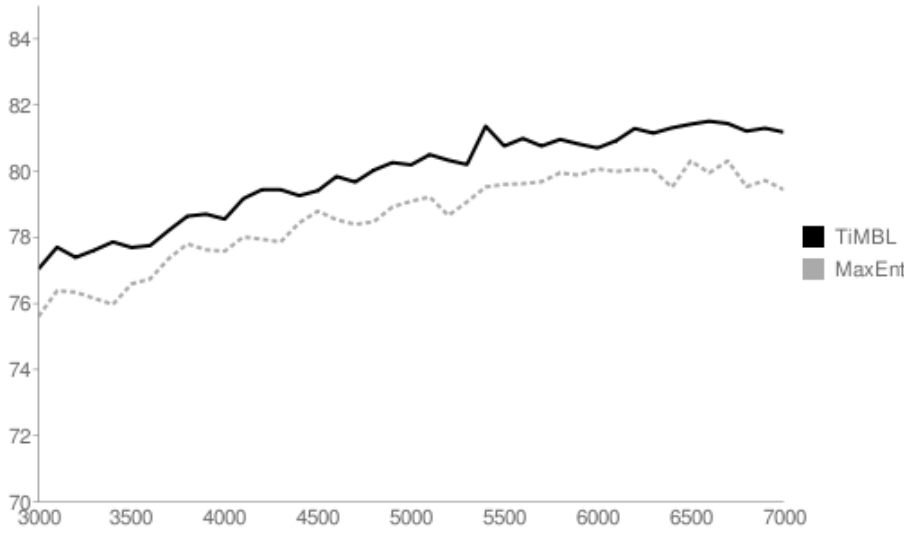

Figure 4.2: Evaluation of the memory-based and the MaxEnt-based named entity recognizers using gradually increasing training corpora, as discussed in section 4.9.7. The horizontal axis shows the corpus size, while the vertical axis displays accuracy.

Chapter 5

Prepositional Phrase Attachment Disambiguation

5.1 Introduction

This chapter deals with the question of whether a prepositional phrase (PP) modifies, or attaches to, the noun in front of it, or if it rather attaches to the main verb of the clause.

Options other than noun or verb attachment exist, of course—a PP could attach to an adjective (*svart* ***som natta*** "black **as the night**") or a pronoun (*vi* ***fra Østkanten*** "we **from the East Side**"). In line with most previous work on automatic PP attachment disambiguation, however, I will focus on those cases where the PP attaches either to the main verb of the clause or to a noun which immediately precedes the PP, since these constitute the vast majority of PP attachment cases.

A few words about terminology are in order. I could use either *attachment* or *modification* to denote the kind of relationship that I am discussing in this chapter. The term *attachment* describes the relationship between a PP and a noun or a verb from a syntactic point of view, while the notion of *modification* describes it from a semantic perspective. Since my approach is neither purely syntactic nor purely semantic—the use of co-occurrence frequencies captures both syntactic and semantic relations without distinguishing between them—I make the more or less arbitrary decision to use the term *attachment* in most cases to denote both the syntactic and the semantic aspect of the relationship.

Correct attachment of PPs is a notorious problem for natural language parsers. This problem typically occurs in sentences like those in (1), where we have a verb followed by a noun phrase (NP) and a PP.

(1) a. Anna saw the man with a telescope.
b. Anna saw the top of the mountain.
c. Anna watched TV for three hours.

In (1-a), the PP *with a telescope* can be considered to attach to either the verb *saw* or the noun *man*[1]. In the first case, the PP tells us that Anna saw the man through her telescope; in other words, it conveys information about how the act of seeing was carried out. In the second case, the PP tells us that the man that Anna saw had a telescope. This is a classical example of syntactic ambiguity, where neither human readers nor automatic systems can determine the correct attachment site without more context (whether textual or non-textual).

Example (1-b), on the other hand, can easily be disambiguated by a human reader; in this case, there is no doubt that the PP *of the mountain* attaches to *top* rather than to *saw*. Likewise, in the case of (1-c), a human reader will have no problem determining that the PP describes the act of watching and therefore attaches to the verb.

To an automatic parser, however, (1-b) and (1-c) are equivalent to (1-a), unless the parser is equipped with the kind of knowledge that is extracted by the techniques described in this chapter. Without any knowledge about the fact that *of* is much more likely to attach to nouns than to verbs, the parser has no way of determining the correct attachment site for the PP in (1-b). Likewise, unless the parser is aware of the fact that the PP *for three hours* is a temporal phrase that normally attaches to the verb of the clause, it will lack the knowledge required to disambiguate (1-c).

Although this task poses a difficult problem for automatic systems, being able to solve it is important for many higher-level language processing tasks, in addition to the task of anaphora resolution as discussed below. One such task is the identification of requested information about entities or actions in natural language, e.g., for information retrieval, information extraction, or question-answering purposes. For example, if the information contained in a certain PP pertains to the noun preceding it, rather than to the main verb of the clause, then it constitutes part of the information that can be extracted for this particular noun.

Another task which greatly benefits from PP attachment disambiguation is that of identifying suitable prosodic boundaries for speech generation (Marsi et al., 1997; van Herwijnen et al., 2003). When a PP is attached to the preceding noun, such as in (1-b), there will normally be no pause or other kind of boundary mark (such as a falling tone) between the noun and the PP in spoken language, and hence a speech

[1]In the machine learning literature, the problem has traditionally been stated as one of choosing whether to attach the PP to the main verb or to the noun (or one of the nouns) following the main verb. In syntactic theory, however, the PP is usually considered to be attached to the VP rather than directly to the verb. An interesting topic for future work would be to investigate the effects of applying an analysis involving VP rather than verb attachment to this task. For the time being, however, I will stick to the traditional machine learning analysis for compatibility with earlier work in this field.

generation system should avoid inserting a boundary mark in order to produce natural sounding speech. Conversely, with verb attachment, as in (1-c), a pause between the noun and the PP will usually be deemed appropriate by native speakers (Marsi et al., 1997).

5.2 Relevance for anaphora resolution

Some researchers (e.g., Lappin and Leass, 1994) have found that an NP which is embedded within another NP seems to be less accessible as an antecedent compared to NPs that are not embedded in such a way. Furthermore, occurring in the complement of a PP is one of the most common ways in which an NP can become embedded within another NP (along with occurring in a relative clause). Hence, the question of whether a PP attaches to a verb or a noun may influence the likelihood that its complement is an antecedent of a following anaphor.

5.3 Disambiguation approach

In this chapter, I present a somewhat novel approach to PP attachment disambiguation, which I first described in Nøklestad (2005). In this approach, unambiguous evidence from a corpus is extracted automatically and used to train a standard supervised learning mechanism, which can in turn be used to disambiguate ambiguous attachments.

In other words, I am applying a kind of "pseudo-supervised" approach. It is supervised in the sense that it involves training a regular supervised learning mechanism (e.g., a memory-based learner or a maximum entropy model) by feeding it pre-classified data examples. On the other hand, the method differs from traditional supervised training in that the pre-classified training data are not exactly the same kind of data that the mechanism will later be required to disambiguate, since the training data are gathered from unambiguous contexts which are different from those contexts in which disambiguation will take place. Furthermore, the training data are created automatically, without the need for the human annotation that normally makes the use of supervised learning mechanisms much more labour-intensive than unsupervised learning.

5.4 Earlier work

Early proposals for resolving PP attachments relied entirely on syntactic information, most notably the principle of Right Association, which states that a constituent tends

to attach to the lowest, rightmost node in the syntactic tree (Kimball, 1973)[2], and the principle of Minimal Attachment, which claims that a constituent tends to attach in a way that requires as few additional syntactic nodes as possible (Frazier, 1979).

However, as pointed out by Hindle and Rooth (1993), these principles make contradictory predictions with regard to the kind of ambiguous attachment situation that is the topic of this chapter. Right Association predicts noun attachment, because the noun is located lower and more to the right in the tree than the verb. Minimal Attachment, on the other hand, predicts verb attachment, because that requires the smallest number of nodes in the tree (Frazier, 1979, p. 26). Thus, each of these principles will only apply in a very limited amount of attachments, and they cannot really be used to select the correct kind of attachment.

In contrast, Whittemore, Ferrara, and Brunner (1990) showed that PP attachment is largely determined by the lexical preferences of specific nouns, verbs, and prepositions. Thus, in order to enable an automatic system to perform PP attachment disambiguation in a more reliable way, we need to provide it with information about the strength of the association between a PP and a verb or a noun. One way of obtaining this kind of information which requires a minimum of manual effort is to extract it from corpora using some kind of machine learning method. Since this is the approach taken in this thesis, I will focus here on earlier work that uses some form of machine learning to solve this task.

A range of statistical or machine learning methods have been applied to this task in the past, some supervised and some (semi-)unsupervised.

The first to make use of information automatically extracted from a corpus were Hindle and Rooth (1993). They selected verb or noun attachment based on the so-called *lexical association* between prepositions and nouns or verbs. Lexical associations were calculated on the basis of frequency counts from 13 million words of newspaper text. Following the semi-unsupervised work by Hindle and Rooth, several supervised PP attachment disambiguators were developed, including systems based on transformation-based learning (Brill and Resnik, 1994), maximum entropy modelling (Ratnaparkhi et al., 1994) and other kinds of log-linear models (Franz, 1996), backed-off maximum likelihood estimates (Collins and Brooks, 1995), memory-based learning

[2]I would like to take this opportunity to correct Hindle and Rooth (1993)'s portrayal of Kimball's Right Association principle, which has also been adopted in various later work due to the seminal status of Hindle and Rooth's paper.

Hindle and Rooth present Kimball's principle in the following way: "a constituent tends to attach to another constituent immediately to its right" (Hindle and Rooth, 1993, p. 103). However, this is clearly not what the principle states. What it states is that a constituent tends to attach to the lowest, rightmost node in the tree. Moreover, in the kind of construction that Hindle and Rooth (as well as the present chapter) are concerned with, the noun is not located to the right of the PP (at least not in the surface structure, nor in the tree structures used in mainstream syntactic theory).

Thus, if Hindle and Rooth's portrayal of the principle were correct, the principle would in fact not apply to this kind of PP attachment at all, at least not in languages with prepositions rather than postpositions (which is of course the case for English, the language that Hindle and Rooth deal with).

(Kokkinakis, 2000; van Herwijnen et al., 2003; Zavrel et al., 1997) and other nearest-neighbour approaches (Zhao and Lin, 2004), decision trees (Stetina and Nagao, 1997), boosting (Abney, Schapire, and Singer, 1999), rule induction (van Herwijnen et al., 2003), and support vector machines (Olteanu and Moldovan, 2005; Vanschoenwinkel and Manderick, 2003).

Although the best performing systems are, as usual, based on supervised learning, considerable work has also been invested in the development of unsupervised or semi-unsupervised systems for this task, in addition to Hindle and Rooth's original system. Ratnaparkhi (1998) used a different semi-unsupervised approach which improved on Hindle and Rooth's results, and Pantel and Lin (2000) achieved results that approached the best supervised systems at the time. See the discussion in section 5.13, however, for important remarks on Ratnaparkhi's and Pantel and Lin's results.

Finally, there has been some work that goes beyond the traditional approaches that involve supervised learning or unsupervised learning from fixed corpora. Volk (2001), van Herwijnen et al. (2003), and Olteanu and Moldovan (2005) make use of co-occurrence counts from the World Wide Web to improve PP attachment in English and Dutch, respectively. Volk (2002) combines a supervised system that has only a limited amount of training data with an unsupervised method, and shows that the combined system outperforms each of the supervised and unsupervised components on their own. Merlo and Ferrer (2006) extend the task to one of making a four-way classification into verb attachment, noun attachment, argument, and adjunct, maintaining that the argument/adjunct distinction is equally important to the interpretation of a PP as the verb/noun attachment distinction.

While I am not aware of any previous data-driven work on Norwegian PP attachment disambiguation, some work has been carried out on Swedish, which is a closely related language. Kokkinakis (2000) combines supervised, memory-based learning with unsupervised extraction of data from an electronic lexicon and from corpora (in the latter case using Hindle and Rooth's heuristic that PPs rarely attach to pronouns). Aasa (2004) applies a modified version of Volk (2001)'s unsupervised method, in which he calculates co-occurrence values for prepositions, nouns, and verbs from training data. Aasa also uses clustering to create synonym classes for nouns in order to reduce data sparseness problems and increase the performance of his system.

As stated above, for my own work I apply a kind of pseudo-supervised approach to this task. Here, as elsewhere in this thesis, I have used memory-based learning as my primary machine learning method, but I have also run experiments in which the memory-based learning mechanism is replaced by a maximum entropy model.

5.5 Relevant constructions

Norwegian is like English in that PP attachment ambiguities arise in cases where the main verb of the sentence is followed by an NP and a PP; the problem consists of determining whether the PP attaches to the noun or the verb.

Consider the following examples:

(2) a. De spiste pizza på Pizza Hut.
they ate pizza at Pizza Hut
"They had pizza at Pizza Hut."

b. De spiste pizza fra Pizza Hut.
they ate pizza from Pizza Hut
"They had pizza from Pizza Hut."

(3) a. De spiste pizza med fingrene.
they ate pizza with fingers-the
"They ate pizza with their fingers."

b. De spiste pizza med pepperoni.
they ate pizza with pepperoni
"They ate pizza with pepperoni."

These examples show that both the choice of preposition and the choice of prepositional complement influence what kind of PP attachment we get. In (2-a) and (2-b), the PP attachments differ due to the different prepositions that are used: in (2-a), the PP attaches to the verb, while in (2-b) it attaches to the noun. In fact, experiments by Collins and Brooks (1995) indicate that the preposition is indeed the most important lexical item for determining PP attachment. However, (3-a) and (3-b) show that there are also cases where we have identical prepositions but different attachment sites due to different prepositional complements—*fingrene* "the fingers" results in verb attachment in (3-a), while *pepperoni* "pepperoni" causes noun attachment in (3-b).

(4) Hansa-guttene håpet å *ta sin første stafettseier i Sandefjord*, men måtte ta til takke med tre sølv etter uimotståelige Varg. (*BT*)
"The Hansa boys were hoping to *take their first relay victory in Sandefjord*, but had to settle for three silver (medals) after invincible Varg".

In (2) and (3), correct attachment could easily be determined by a native speaker. Example (4), on the other hand, is an authentic example, taken from the newspaper *Bergens Tidende* (BT), of a sentence with the kind of genuine ambiguity that was previously illustrated by the artificial example in (1-a) (my italics emphasize the relevant part of the sentence). The sentence could either mean that, in Sandefjord (a Norwegian town), the Hansa boys were hoping to take their first victory *ever* (implying verb attachment), or it could mean that, after having lost one or more relay runs in

Sandefjord, they were now hoping to take their first victory there (noun attachment). In order to determine the correct attachment site, we either need a wider linguistic context or prior knowledge about whether or not this team has ever taken a relay victory.

Finally, as noted by Hindle and Rooth (1993) for English, it is surprisingly common to find cases where it is difficult to make an attachment decision, regardless of context size and the amount of background knowledge we have. Such cases include idiomatic expressions, light/support verb constructions, and cases of vagueness. In the following sections, I will have a brief look at these problematic cases and discuss how to deal with them.

5.5.1 Idiomatic expressions

In idiomatic expressions, the meaning of the whole expression cannot be deduced from that of its parts. An example is given in (5), taken from the newspaper *Verdens Gang* (VG).

(5) Ta rotta på sjefen (*VG*)
take rat-the on boss-the
"Outsmart your boss"

In cases like this, the meaning of the verb–noun–preposition sequence cannot be decomposed into a combination of the meanings of each of these separate components. Therefore, this sequence is most suitably viewed as a single semantic unit, and the preposition cannot be said to be attached to either the verb or the noun—rather, all three of these elements are tightly attached to each other. Unfortunately, the traditional analysis of PP attachment in the machine learning literature does not permit simultaneous attachment. Hence, if we want to retain this kind of analysis, we are forced either to make a more or less unmotivated decision, or to disregard the example altogether.

5.5.2 Light verb constructions

Another relevant group of constructions is constituted by *light verb constructions* (sometimes called *support verb constructions*) (Sag, Baldwin, Bond, Copestake, and Flickinger, 2002). This kind of construction is exemplified in (6).

(6) a. Han tok en spasertur i parken.
he took a walk in park-the
"He took a walk in the park."

b. Han tok en spasertur etter middag.
he took a walk after dinner
"He took a walk after dinner."

In light verb constructions, the verb (as the name indicates) has very little semantic content, and the bulk of the meaning of the verb phrase (VP) is located in the noun. This fact seems to be what led Hindle and Rooth (1993) to consistently attach the PP to the noun in such cases. However, such an attachment decision is often questionable, for instance in cases where the PP functions as a temporal adverbial, as can be seen by comparing (6-a) and (6-b).

In (6-a), one might possibly argue, as Hindle and Rooth (1993) do, that the PP attaches most naturally to the noun, although the correct decision is far from clear[3]. In (6-b), on the other hand, the PP conveys information about the time of the event, thus making it highly unnatural to attach it only to the noun—it is, after all, the verb that expresses the occurrence of the event. In general, although different light verb constructions might seem to favour noun or verb attachment to varying degrees, it could be argued that the PP really attaches to the entire complex predicate and therefore again cannot be said to attach to either the verb or the noun[4].

(7) a. Han fører krig mot Tsjetsjenia. (*Bergens Tidende* 1995)
he wages war towards Chechnya
"He wages war on Chechnya."

b. ?Mot Tsjetsjenia fører han krig.
towards Chechnya wages he war
"On Chechnya he wages war."

c. ??Krig mot Tsjetsjenia fører han.
war towards Chechnya wages he
"War on Chechnya he wages."

d. Tsjetsjenia fører han krig mot.
Chechnya wages he war towards
"Chechnya he wages war on."

Some indication about attachment sites for PPs can be found by investigating their topicalization possibilities. If the PP can be topicalized by moving it away from the noun (as in (7-b)), it is an indication that the PP functions as a separate clause constituent (usually an ADVERBIAL) which attaches to the verb. If, on the other hand, the PP can only be moved together with the noun (as in (7-c)), we have indications of noun attachment. Although topicalization constitutes only one type of information

[3]It should be noted that Hindle and Rooth (1993) are aware that their decisions might be questionable, and that they explicitly state that these decisions should not be taken as authoritative.

[4]If the task had been formulated in a way which involved a choice between noun and *VP* attachment rather than noun and *verb* attachment, (6-b) could have been seen as an unproblematic case of VP attachment. With such a task formulation, we could state that, from a syntactic point of view, the PP attaches to the VP regardless of whether it modifies the semantics of the whole VP or that of the verb alone. Although I adopt the distinction between noun and verb attachment which is prevalent in the machine learning literature, I am nevertheless influenced by this alternative approach in that I choose verb attachment in cases like (6-b) when creating the training and test corpora (cf. section 5.6).

and cannot be taken as conclusive evidence for one attachment type or the other, it does provide some important clues.

The close association of the PP to both verb and object in light verb constructions is supported by the fact that topicalization of either the PP alone (as in (7-b)) or of object NP + PP (as in (7-c)) yields constructions that seem slightly less natural than movement of the prepositional complement only (as in (7-d))[5]. The use of (7-b) would indicate that the PP does not attach to the noun (since the the PP can move away from it), while the use of (7-c) suggests lack of verb attachment (since the PP moves together with the noun, making the two act as a unit in which the PP is attached to the noun). Since neither of these is as natural sounding as (7-d), the examples indicate that both verb and noun are modified to some degree by the PP.

Compare (7) with (8), where the PP clearly modifies the noun, and hence topicalization of noun + PP is most natural, and with (9), where the PP unambiguously modifies the verb, and consequently topicalization of only the PP sounds best.

(8) a. De spiste pizza fra Pizza Hut i går.
"They had pizza from Pizza Hut yesterday."
b. *Fra Pizza Hut spiste de pizza i går.
"From Pizza Hut they had pizza yesterday."
c. Pizza fra Pizza Hut spiste de i går.
"Pizza from Pizza Hut they had yesterday."

(9) a. De spiste pizza på Pizza Hut i går.
"They had pizza at Pizza Hut yesterday."
b. På Pizza Hut spiste de pizza i går.
"At Pizza Hut they had pizza yesterday."
c. *Pizza på Pizza Hut spiste de i går.
"Pizza at Pizza Hut they had yesterday."

5.5.3 Simultaneous modification

(10) ...som hver vinter *arrangerer skirenn på Fløyen.* (*Bergens Tidende* 1995)
"...who every winter *arrange a ski race at Fløyen.*"

(11) ...og etter kort tid fikk hun tilbud om å *kjøpe en ny gård i samme område.* (*Bergens Tidende* 1995)
...and soon she was offered to *buy a new farm in the same area.*

Examples (10) and (11) illustrate yet another problematic, but rather common construction type, viz. cases of simultaneous modification, or vagueness. When a ski race

[5]The English translation of (7-d) is not very well formed because of restrictions on prepositional stranding that do not apply to Norwegian.

is arranged, both the act of arranging the event (or at least the part of the arrangement act that takes place during the event) and the event itself occur at the location denoted by the PP, and thus the PP seems to modify both the verb and the noun to a certain extent (Schütze, 1997).

Alternatively, one might argue that the modification is ambiguous rather than vague, i.e., that the person expressing the sentence does in fact intend the PP to modify either the noun or the verb, but that the actual attachment site cannot be deduced by the receiver and that it is inconsequential for his/her interpretation of the sentence.

Regardless of the preferred analysis, this is another case where one might argue that no attachment decision should be made. The main difference between cases of simultaneous modification, on the one hand, and idiomatic and light verb constructions, on the other, is that in the case of simultaneous modification, the verb and the noun do not really form a complex predicate—rather, they remain two separate constituents.

5.6 Treatment of unclear attachment cases

As shown in the previous sections, there are several types of ambiguous constructions that make it hard even for human annotators to make a principled decision between verb and noun attachment. It is not easy to determine the number of difficult cases in the gold-standard corpora used in the present work, since there were large differences in the degree of difficulty: the corpora exhibited a continuum ranging from completely clear-cut cases to instances where it was felt that no non-arbitrary decision could be made, and it is hard to draw a line between clear-cut and problematic cases. Still, since I wanted to be able to compare the results of the system against those of earlier systems developed for other languages, a decision was made in each case in which one of the alternatives felt in some way more natural than the other, while the remaining cases were excluded from the corpora.

Using this strategy of trying to retain as many cases as possible resulted in only about 5% of the cases being left out (cf. Table 5.1 on page 109, where these are listed as the *unresolved* cases). Nevertheless, the abundance of more or less difficult cases questions the validity of a performance measure based on a comparison between the output of the system and a manually annotated corpus. Hence, it could be argued that a qualitative evaluation of a system that includes the PP attachment disambiguator as one of its components (e.g., an information extraction system or a dialogue system) would ultimately provide a more informative evaluation. In such a setting, we would evaluate the ability of the system to extract relevant information about events and

entities denoted by the verbs and nouns in the text. If a PP does in fact modify (and hence provide some information about) both a verb and a noun, the system would be credited rather than discredited for extracting that information.

Having pointed out that simultaneous modification does occur (and quite frequently at that), in many light verb constructions or cases of simultaneous modification the PP may nevertheless be felt to be more strongly associated with either the verb or the noun. In (7) on page 104, the PP does after all seem to be most strongly associated with the verb (at least according to my own linguistic intuition), an interpretation that is supported by the fact that (7-b) sounds somewhat better than (7-c). Since the present system (like all other systems I am aware of) is restricted to determining a single attachment site for each preposition, the gold-standard corpora used for testing and development were also restricted to one attachment site per preposition, with the most strongly associated site selected in cases of simultaneous modification.

Although a representation indicating (degrees of) simultaneous modification might have reflected the properties of the language more accurately, the desire to apply PP attachment disambiguation to traditional parses (e.g., for use in a treebank) necessitates a definite decision in each case. Furthermore, the subtleties of simultaneous modification, as perceived by humans, are likely to involve semantic and pragmatic factors that would in any case be unavailable to an algorithm trained on simple sequences of the kind employed here, and hence I would not expect to see these subtleties reflected in the disambiguation model to any significant degree.

Another point to note is that different annotation guidelines may lead to different performance levels for automatic attachment systems. For example, as mentioned above, Hindle and Rooth (1993) acknowledge the problem of assigning attachment sites in light verb constructions, where the semantics of the VP cannot be easily decomposed into a combination of the semantics of the verb and the noun. Hindle and Rooth's solution is to treat all light verb constructions as cases of noun attachment, "based on the fact that it is the noun that provides the lexical information about what prepositions are possible" (Hindle and Rooth, 1993, pp. 14–15).

Although it might well be intended as a way of making consistent decisions for light verb constructions, this guideline seems to favour the decisions that the system will make for such constructions, since it is based on the availability of the kind of lexical information that is provided to the system. Furthermore, as pointed out in section 5.5.2, this guideline might lead to counterintuitive decisions in cases like (12) on page 108, where the PP functions as a time adverbial that seems more closely attached to the verb than to the noun—or perhaps rather to the whole verb+NP complex, in other words, to the VP. In the present work, all decisions are based on the intuitions of the annotator, and in cases where the PP is perceived as modifying

the semantics of the whole VP, verb attachment (interpreted as VP attachment) is selected.

(12) John took a walk after dinner.

While Hindle and Rooth's treatment of light verb constructions might possibly present their disambiguator with an advantage due to the focus on richer noun semantics, the guidelines used in the present project might be advantageous for the disambiguator in another respect. The possibility of topicalization of the PP plays an important part in determining the correct attachment site, and topicalized PPs are part of the unambiguous evidence that the disambiguator makes use of, in that topicalized PPs are considered to exemplify cases of unambiguous PP attachment. I will argue, however, that this criterion is more in concordance with widespread syntactic theories, and as such is motivated independently of the requirements of the disambiguator.

Although most cases can be manually disambiguated in the end, there are some cases in which a decision between verb and noun attachment would seem entirely arbitrary. This is especially true for some idiomatic expressions such as *ta stilling til* "take a stance on/decide on" and *ta kontakt med* "contact" (lit. "take contact with"). Such cases were left unannotated in the gold-standard corpora.

Note that, in the current system, a PP is restricted to attach to the closest noun to the left; other possible noun attachments are not considered. In the overwhelming majority of cases, this restriction will yield the correct result, but it will fail in some cases. In (13), for instance, the PP *om at jeg ikke hadde lov til å si nei* attaches to *beskjed* rather than *Gullestad*, which means that this construction cannot be handled by the system. Although this type of construction is not too uncommon, it is still worth imposing the restriction, since it works in almost all cases and leads to a great reduction of possible choices for the disambiguator.

(13) Jeg fikk beskjed av Anne Gullestad om at jeg ikke hadde lov
I got message from Anne Gullestad about that I not had permission
til å si nei. (*Bergens Tidende* 1995)
to-prep to-inf say no
"I was told by Anne Gullestad that I was not allowed to say no."

5.7 Training, development, and testing corpora

My automatic disambiguator has been trained and tested on the *bokmål* part of the Oslo Corpus of Tagged Norwegian Texts (cf. chapter 3).

One thousand clauses containing ambiguous V-NP-PP contexts, taken from the newspaper *Bergens Tidende*, have been extracted for use as a development corpus, allowing me to study the types of mistakes that the system makes in order to continu-

ously improve the system. Another 1000 clauses from the same newspaper have been extracted for use as a pure test corpus. Some of the clauses contain more than one ambiguous context, resulting in a total of 1010 cases in the development corpus and 1002 in the test corpus.

While creating the corpora, I noticed that, occasionally, the same newspaper story would appear in several issues of the same newspaper within a short period of time. Thus, in order to reduce the risk of having overlapping material between the training corpus and the development and test corpora, the rest of the *Bergens Tidende* material from the same year was discarded. The remainder of the corpus was used for training.

Table 5.1 shows the number of unresolved, excluded, and annotated cases in each corpus, and specifies the number of verb and noun attachments among the annotated cases.

	Total	Unresolved	Excluded	**Annotated**	**V**	**N**
Dev.	1010	53	79	**878**	**384 (43.7%)**	**494 (56.3%)**
Test	1002	49	59	**894**	**384 (43.0%)**	**510 (57.0%)**

Table 5.1: Selected information concerning the development and test corpora, showing the number of unresolved cases, the number of cases that were excluded due to errors in the input, the number of cases that were annotated, and the number of verb attachments and noun attachments, respectively, among these annotated cases. Only figures in bold were used in the calculations of the performance of the system.

Only cases for which the annotators were able to make a decision were used in the calculation of performance accuracy of the disambiguator (see sections 5.5 and 5.6 for a discussion of the difficulties involved in determining correct attachment sites); the remaining cases are listed as *unresolved* in Table 5.1[6]. Furthermore, a number of cases were excluded due to typographical errors, mistakes in tokenization, use of the *nynorsk* norm, or incorrect grammatical tagging. These are listed in the *excluded* column of Table 5.1.

The latter category also includes constructions involving an intransitive or stranded preposition followed by an NP, since in those cases the NP is in fact not a prepositional complement. However, if this preposition was followed by one or more transitive prepositions, the whole sequence was regarded as a complex preposition and included in the performance calculations, unless the intransitive or stranded preposition clearly had to be considered as a separate constituent. Examples of the first type are shown in (14) and (15), where the italicized parts show the complex prepositions. Example (16) shows an instance of the second type, where *på* is a stranded preposition and as such cannot be regarded as part of the following PP.

[6]The development corpus was only tagged by me, while the test corpus was tagged separately by two annotators: myself and Lars Nygaard.

(14) Mandag 4. januar 1954 kommer Elvis *tilbake til* Memphis Recording
Monday 4 January 1954 comes Elvis back to Memphis Recording
Service for å lage en ny privat plateinspilling. (*Bergens Tidende* 1995)
Service for to make a new private record recording
"On Monday 4 January 1954, Elvis returns to Memphis Recording Service to make a new, private recording."

(15) Det er første gang Gjøvik-gutten har tatt steget *opp på* seierspallen i
it is first time Gjøvik-boy-the has taken step-the up on podium-the in
et enkeltrenn i verdenscupen.
a single-race in world-cup-the
"This is the first time the boy from Gjøvik has made it onto the podium in a single race in the World Cup."

(16) Dette er bare noen av de spørsmålene vi får svar *på i* programmet.
this is only some of the questions-the we get answer on in show-the
"These are only some of the questions that are answered in the show."

The development and test corpora are HTML formatted with different attachments having different Cascaded Style Sheets (CSS) classes, which makes it easy to display the corpora in a web browser with different attachment types indicated in different styles (e.g., in different colours). Figure 5.1 on page 128 repeats the formatted corpus example that was shown in chapter 3.

5.8 Training and testing

As mentioned at the beginning of this chapter, the work described here uses a machine learning approach that, to my knowledge, has not been applied before.

On the one hand, each input (a vector of lemmas in a particular context) is paired with a target output (a value indicating either verb or noun attachment), and in this sense, the method satisfies the normal requirements for supervised learning (see, e.g., Hastie, Tibshirani, and Friedman, 2001).

On the other hand, the target outputs are determined by an automatic algorithm rather than by human annotators, and may therefore contain a larger amount of erroneous examples (although manual inspection indicates that the algorithm has a very high precision). Perhaps more importantly, the training examples are not taken from the ambiguous contexts in which the system will perform disambiguation, since they have to be extracted from *un*ambiguous contexts. For this reason, I prefer to call this a *pseudo-supervised* training regime[7].

The training process consists of the following stages:

[7] I could also have used the term *semi-(un)supervised* learning, but that seems to be more commonly used for training regimes in which the experimenter provides the learner with a few initial examples which are used for bootstrapping the algorithm (Abney, 2002, 2004).

- *Stage 1:* Apply an algorithm that extracts cases of unambiguous PP attachment from the training corpus. Unambiguous verb attachments are found where the PP is either clause initial (17-a) or immediately following the main verb (17-b)[8]. On the other hand, PPs that follow a (base) NP in preverbal position constitute cases of unambiguous noun attachment (17-c).

  (17) a. Etter en stund fant jeg boka.
  after a while found I book-the
  "After a while I found the book."

  b. Hun spiste på restaurant.
  she ate on restaurant
  "She ate at a restaurant."

  c. Boka på bordet er min.
  book-the on table-the is mine
  "The book on the table is mine."

- *Stage 2:* Go through all ambiguous V-NP-PP sequences in the training corpus. Mark the sequence with verb attachment if we found any examples of this verb being unambiguously modified by the PP in Stage 1. Do the same thing for noun attachment. Note that the sequence will be marked with both types of attachment if both unambiguous verb attachment and unambiguous noun attachment were observed in Stage 1. This stage produces a total of 324,440 training instances to be used in the next stage, 124,506 (38.4%) of which are verb attachments and 199,934 (61.6%) are noun attachments. Note that this distribution of verb and noun attachments is fairly similar to the distributions that were shown in table 5.1 for the development corpus (V: 43.7%; N: 56.3%) and the test corpus (V: 43.0%; N: 57.0%), but the training material contains a somewhat larger portion of noun attachments.

- *Stage 3:* Train a machine learner on these marked-up sequences.

Finally, testing the system (or applying it to new text) amounts to querying the machine learner for the attachment type of each ambiguous context.

5.8.1 Machine learning features

As is usual in machine learning, we need to decide on a number of features which will represent the data when they are fed into the machine learning mechanism, and which will be weighted differently so as to reflect their respective importance for the task at

[8]This kind of construction might seem irrelevant at first, since it does not involve a postverbal (OBJECT) noun. However, *spise* "eat" is one of many semi-transitive verbs, which might occur with or without an object, and statistics from this kind of construction are valuable for determining attachment in cases where the verb occurs transitively.

hand. As in various earlier work, such as Ratnaparkhi et al. (1994) and Zavrel et al. (1997), the features include the lemmas of each of the "head words", i.e., the verb (V), the head noun of the NP (N), the preposition (P), and the head of the prepositional complement (N2). The system also includes compound analysis of N and N2, and the heads of the compounds (or the entire nouns, if non-compound) are included as additional features, giving a total of six input features representing each ambiguous context.

Table 5.2 illustrates the use of a trained memory-based learner employing these features to classify a test instance. The test instance is extracted from (1-b) on page 98 (*Anna saw the top of the mountain*), and the example uses (fictitious) training instances extracted from the following sentences:

- From the plane I saw the roof of my house.
- He reached the top of the ladder.
- They pulled the horse sleigh across the mountain.

	V	N	P	N2	N head	N2 head	Category
Train 1	**see**	roof	**of**	house	roof	house	N
Train 2	reach	**top**	**of**	ladder	**top**	ladder	N
Train 3	pull	horse sleigh	across	**mountain**	sleigh	**mountain**	V
Test	see	top	of	mountain	top	mountain	*N*

Table 5.2: Classification based on nearest neighbours. Feature values in bold indicate a match with the test instance.

Table 5.2 displays the feature values in the test instance in the bottom row, and those of the training instances in the rows above. Training instance values which match the values of the corresponding features in the test instance are shown in bold.

Training instances 1 and 3 match the test instance on two feature values, while training instance 2 matches on three. Hence, training instance 2 is the nearest neighbour to the test instance, and since its category is *N*, the test instance will also be (correctly) assigned to the *N* category.

5.9 Compound analysis

As mentioned in section 5.8.1, the results of compound analysis figures among the features used for this task. Since no compound analyser for Norwegian was readily available for my purposes at the time I carried out my experiments, I created my own mechanism for compound analysis.

It should be pointed out that the Oslo-Bergen tagger contains a more sophisticated compound analyser that has recently been made available for standalone use. At the

time this work was carried out, however, it was not available in an accessible form. Furthermore, although the analyser presented here is more simplistic than the one included in the Oslo-Bergen tagger, it has the advantage that it works very efficiently due to its simple algorithm and its use of analysis caching, a property which makes it well suited for inclusion in tools for real-time processing of language.

Although the analyser presented here is different from the one used by the Oslo-Bergen tagger, it uses the same lexicon to check that the compound parts it finds are actual lemmas. The goal of the analyser is to identify a probable head for the whole compound, not to find the most probable compound structure as a whole. In other words, it is not a generalized compound analyser, but rather a specialized tool for the purposes of head extraction. Nevertheless, all parts of the proposed analysis are checked against the lexicon, not only the head.

Processing starts at the end of the compound lemma and conducts recursive compound analysis with backtracking at each recursion level, all the time trying to find the smallest possible head (assumed to be the right part) of each compound or subcompound. The system handles compounding morphs (such as the formatives found in *barn**e**hage* "kindergarten" and *dag**s**tur* "day trip"). However, it does not consider potentially relevant linguistic information such as the part-of-speech of the first part of the compound or the identity of the characters at the end of that part, because creating an analyzer with that level of linguistic sophistication was considered to be beyond the scope of the present work.

The system implements the following algorithm for analysing a potentially compound lemma:

1. Start with the whole lemma
2. If the lemma is found in our *compound cache*, return the compound and exit
3. Look for the smallest rightmost subpart of the lemma that can be found in the lexicon, restricted to lemmas containing at least 3 characters; if found, this will be considered the compound head
4. If no head was found, consider the lemma to be non-compound and exit
5. If a head was found, is the remainder of the lemma itself a lemma?
    5.1. If yes, consider the compound successfully analysed and exit
    5.2. If no, try to chop off an infix (s or e). Is the remainder now a lemma?
        6.1. If yes, consider the compound successfully analysed and exit
        6.2. If no, go to step 2 and try to analyse the remainder as a compound (either including or excluding any infix we may have found in 5.2)

Distance metric	Metric threshold	Weighting scheme	k	Extrapolation method	Accuracy on optimization data
MVDM	1	None	9	ID	78.10

Table 5.3: Optimized parameter settings for TiMBL and the accuracy reported by Paramsearch on the optimization data. Note that this accuracy does not reflect a proper evaluation of the system; see Tables 5.5 and 5.6 for proper evaluations on independent test data. MVDM = Modified Value Difference Metric; ID = inverse distance weighting.

	V	N	N head	P	N2	N2 head
Number of different values	5189	46954	18097	1216	56549	23531
Information gain	0.12	0.35	0.23	0.09	0.23	0.12
Gain ratio	0.02	0.03	0.02	0.02	0.02	0.01

Table 5.4: Gain ratio weights used by TiMBL in the experiments with default settings. V = verb; N = OBJECT noun; P = preposition; N2 = prepositional complement.

All compounds that are found are registered in the cache, which is checked at stage 2 of the algorithm.

Of course, the length of a compound must be at least twice the minimum lemma length that we allow. Since the minimum lemma length was set to 3, it means that we only consider words of at least 6 characters for analysis. Furthermore, when we get to step 5.2, the lemma length is checked again to see that it has not sunk below 6 characters before the mechanism jumps to step 2. If it has, then we know that we cannot get an acceptable analysis, and the word is classified as a non-compound.

In order to exclude lemmas that should not be allowed as possible heads of the whole compound (perhaps because they are homonymous with common suffixes), they can be marked as being unsuitable as compound heads. This will force the algorithm to search for longer heads.

5.10 Automatic parameter optimization

The 324,440 training instances representing unambiguous attachment that were produced in stage 2 of the procedure described in section 5.8 were used to train a memory-based learner. Paramsearch (van den Bosch, 2004, cf. section 3.7) was applied to the training data in order to optimize the parameters of the learning algorithm. The optimized parameter values are shown in Table 5.3. In accordance with Zavrel et al. (1997), Paramsearch finds the MVDM similarity metric and inverse distance weighting (Dudani, 1976) to be good choices for this task. More surprisingly, the optimal

weighting scheme is found to be no weighting at all. In other words, all features should be treated as equally important.

This does make sense, in fact, when looking at the information gain and gain ratio weights that are computed by TiMBL and listed in Table 5.4. As shown in the second row of the table, the information gain values for the various features are quite different. However, when the information gain values are adjusted for the number of different values that the feature takes, displayed in the first row, we get the (usually more valuable) gain ratio values as shown in the bottom row. We can see that the gain ratio values are very similar for all of the features, making it plausible that, in this particular case, gain ratio weighting does not improve performance.

The finding that the features should not be weighted differently contrasts with the work by Collins and Brooks (1995) on English PP attachment. Collins and Brooks find that the preposition is the most important element, based on huge drops in accuracy that are found when the preposition is left out of their vectors (much bigger than when other elements are left out). Zavrel et al. (1997), also working on English, corroborate this by using an information gain weight for the preposition of 0.1, which is about three times the weight that is given to the other elements (0.03). van Herwijnen et al. (2003), on the other hand, come to the same conclusion in their experiments on Dutch that is reached here for Norwegian; the authors experiment with different weighting schemes, and indeed find that using no weighting is the optimal choice.

These results may be lead us to hypothesize that the identity of the preposition is in fact more important in English than it is in Norwegian and Dutch. However, a proper evaluation of such a hypothesis would require us to handle two or more of these languages within one and the same system in order to ensure that they were treated in the same way. Furthermore, gold standards would have to be created for each of the languages using the same set of annotation guidelines. Such a project is considered to lie beyond the scope of the present thesis.

In fact, the most plausible explanation for the lack of importance of the preposition in the present system can be found by looking at the data shown in Table 5.4. In the first row of the table, we see that there are a surprisingly large number of different prepositions in the training data (1216 different ones). This large number stems from a fair amount of "complex prepositions" (cf. section 5.7), i.e., sequences of two or more prepositions. Some of these seem legitimate, such as *opp+på* "up onto" and *inn+i* "into", some may be better understood as verbal particle + preposition (and should ideally not have been included in my training and test corpora), e.g., *Han var* ***med+i*** *en sekt* "He was part of a cult", lit. "He was **with+in** a cult", and some are plain errors, typically consisting of a stranded preposition followed by a PP (e.g., *Han ble tatt hånd* ***om+av*** *politiet* "He was taken care **of+by** the police").

	Orig. devel.	Orig. test	Reduced devel.	Reduced test
No. of cases	878	893	734	701
% N-attach	56.2	57.0	57.9	58.2
N-attach $F_{\beta=1}$	75.62	77.47	78.86	80.23
V-attach $F_{\beta=1}$	64.43	64.66	68.91	68.63
Overall accuracy	71.07	72.48	74.83	**75.75**

Table 5.5: Results of PP attachment disambiguation experiments with TiMBL using default settings. The table shows the number of test cases in each corpus, the percentage of cases that were manually tagged as noun attachments, F-scores for noun and verb attachments, and overall accuracy. See the text for an explanation of the distinction between original and reduced corpus versions.

	Orig. devel.	Orig. test	Reduced devel.	Reduced test
No. of cases	878	894	734	701
% N-attach	56.3	57.0	57.9	58.2
N-attach $F_{\beta=1}$	75.71	73.03	78.88	74.76
V-attach $F_{\beta=1}$	65.94	58.96	70.47	61.37
Overall accuracy	71.64	67.45	75.37	**69.47**

Table 5.6: Results of PP attachment disambiguation experiments with TiMBL using optimized settings. The table shows the number of test cases in each corpus, the percentage of cases that were manually tagged as noun attachments, F-scores for noun and verb attachments, and overall accuracy. See the text for an explanation of the distinction between original and reduced corpus versions.

These distinctions cannot be detected by the automatic instance extraction mechanism (note that the Oslo-Bergen tagger does not distinguish between prepositions and verbal particles and hence cannot help reduce the number of apparent prepositions by removing particles). However, a manual cleanup of the material might have improved the purity of the training data and thereby increased the gain ratio weight of the preposition feature by reducing the number of values it could take.

5.11 Results

Table 5.5 shows the results of the PP attachment disambiguation experiments using the default parameter settings of TiMBL, while Table 5.6 reports on evaluation using the optimized parameter values found by Paramsearch. The percentages of noun attachments in the manually annotated corpora are included as a baseline, indicating the performance level we would achieve if the system always selected noun attachment. In all tests, the TiMBL classifiers score significantly above this baseline ($p \ll 0.01$).

One result that is particularly clear from these tables is that, unlike the named entity recognition work described in chapter 4, the present task does *not* benefit from parameter optimization. On the test corpora, the classifier with optimized parameters performs significantly worse than the classifier with default settings ($p \ll 0.01$). While I do not have a good explanation for this particular finding, it might have something to do with the fact that the training data (on which the parameter settings were optimized) are not exactly of the same kind as the test data (i.e., unambiguous vs. ambiguous contexts; cf. section 5.8), so that adjusting the parameters too closely to the characteristics of the training data may be detrimental to the performance of the classifier on ambiguous data. In any case, van den Bosch (2004) also found a few cases where automatic parameter optimization led to poorer results than using the default parameter settings of the algorithm, so this situation is not unprecedented.

Focusing on the results with default settings that are shown in Table 5.5, the system initially obtained accuracy figures of 71.07% on the development corpus and 72.48% on the test corpus. Manual inspection of the ambiguous data annotated in Stage 2 of the training process shows that cases which are annotated with both noun and verb attachment in the training data tend to be cases that are also difficult for humans to classify. In other words, for such difficult cases, the training data often provide evidence for both types of attachment. This suggests that these are actual cases of simultaneous attachment, in which case it might not be desirable to force either the human annotators or the automatic disambiguator to make a choice between attachment types.

In order to investigate this further, I have created versions of the corpora that only include cases where the attachment site is considered indisputable; these are the reduced corpus versions in Tables 5.5 and 5.6. As the tables show, this means removing a substantial number of cases from the original corpora. When these cases are removed, overall accuracy on the test corpus increases to 75.8%, which is identical to the accuracy of Hindle and Rooth (1993)'s system for English[9]. Furthermore, I have found that the error rate on the excluded cases in the original corpus was about 40%, corroborating the impression that these cases were difficult for both humans and machines. Although a different choice of guidelines for resolving vague cases might have led to other results, this is nevertheless an indication that such contexts are more difficult for the system to resolve than clear-cut cases, and as such the system mirrors the difficulties that humans have.

[9]Note, however, that Hindle and Rooth did not remove unclear cases, but made a decision in every single case. For idioms and vague cases, they relied on their own intuitions without following any principled guidelines. For light verb constructions, they always attached the PP to the noun. As I have argued earlier in this chapter, I believe that the latter guideline yields incorrect attachments in many cases, and hence it was not used for the present system. Had it been used, it might have made the system look better than it actually is, since the classifier always performs better on noun attachments than on verb attachments.

Parameter estimation method	Gaussian prior	Accuracy on optimization data
L-BFGS	1	74.01

Table 5.7: Optimized parameter settings for MaxEnt and the accuracy reported by Paramsearch on the optimization data. Note that this accuracy does not reflect a proper evaluation of the system; see Tables 5.8 and 5.9 for proper evaluations on independent test data.

5.12 Replacing MBL with MaxEnt and SVM

Although the main focus of this chapter is on using memory-based learning for the PP attachment disambiguation task, it is always interesting to see whether the results change when we use different machine learning methods. With the method of pseudo-supervised training presented here, it is particularly easy to replace the memory-based learner with other vector-based supervised machine learning methods. So far, I have tested the effect of replacing the memory-based learner by a maximum entropy (MaxEnt) model (cf. section 2.3) and a support vector machine (SVM) (cf. section 2.4).

These alternative machine learning methods are trained using the same set of features and the same training data that were used with the memory-based learner. They are trained and tested with their default parameter settings as well as with optimized settings obtained by Paramsearch.

5.12.1 Maximum entropy modelling

The optimized MaxEnt settings and the accuracy of the MaxEnt model on the optimization data as reported by Paramsearch are shown in Table 5.7. Results on the development and test data are displayed in Table 5.8 for the experiments with default settings, and in Table 5.9 for those using automatically optimized parameter settings.

There are two main results that can be discerned from the MaxEnt experiments. First, with this machine learning method, automatic parameter optimization no longer gives a significant performance decrease; now it only leads to a non-significant lowering of the accuracy on the reduced test corpus, from 75.75% to 75.46%. As pointed out by van den Bosch (2004), learning mechanisms that depend on many parameter settings are naturally more likely to benefit from parameter optimization than those that depend on fewer settings. Thus, it is not surprising that the effect of parameter optimization is smaller for the MaxEnt model (with two settings being optimized) than for the memory-based learner (for which five different settings are optimized).

The second main result of these experiments is that the overall accuracy scores of the best MaxEnt model and the best memory-based learner (in both cases, those that use the default settings) on the reduced test corpus are exactly the same. Thus, for this task, the choice between these two machine learning methods cannot be made based on performance level. However, since the MaxEnt model can annotate new data

	Orig. devel.	Orig. test	Reduced devel.	Reduced test
N-attach $F_{\beta=1}$	76.02	75.87	79.44	79.67
V-attach $F_{\beta=1}$	66.30	64.45	70.94	69.96
Overall accuracy	71.98	71.25	75.92	**75.75**

Table 5.8: Results of PP attachment disambiguation experiments with MaxEnt using default settings. The table shows F-scores for noun and verb attachments as well as overall accuracy. See the text for an explanation of the distinction between original and reduced corpus versions.

	Orig. devel.	Orig. test	Reduced devel.	Reduced test
N-attach $F_{\beta=1}$	76.38	75.80	80.05	79.52
V-attach $F_{\beta=1}$	66.57	64.27	71.71	69.40
Overall accuracy	72.32	71.14	76.60	**75.46**

Table 5.9: Results of PP attachment disambiguation experiments with MaxEnt using optimized settings. The table shows F-scores for noun and verb attachments as well as overall accuracy. See the text for an explanation of the distinction between original and reduced corpus versions.

considerably faster than the memory-based learner, MaxEnt might be the best choice in this case.

5.12.2 The support vector machine

Table 5.10 shows the optimized parameter settings and the accuracy on the optimization data for the SVM. The performance of the SVM on development and test data is shown in Table 5.11 for default settings and in Table 5.12 on page 121 for the settings that are determined by Paramsearch.

The accuracy obtained for the SVM on the optimization data (76.29%) lies between that of the memory-based learner (78.10%) and the MaxEnt model (74.01%). On the development and test data, however, the performance of the SVM is abysmal; its accuracy on the reduced test corpus is only 52.94% with default settings and 53.35% using the settings obtained by Paramsearch.

In other words, the SVM does a surprisingly poor job of generalizing from the unambiguous cases to the ambiguous ones, scoring well below the simple baseline of 58.2% that was shown in Table 5.5, which results from always selecting noun attachments. This was unexpected, considering the good performance that has been obtained by SVMs on various other tasks, such as shallow semantic parsing (Pradhan et al., 2005), chunking (Kudo and Matsumoto, 2001), dependency structure analysis (Kudo and Matsumoto, 2000), and named entity recognition (Isozaki and Kazawa, 2002; Mayfield et al., 2003; McNamee and Mayfield, 2002). At the present time, I can-

Training error/margin trade-off	Cost factor	Kernel function	γ	Accuracy on optimization data
5	1	Radial basis	0.064	76.29

Table 5.10: Optimized parameter settings for SVM and the accuracy reported by Paramsearch on the optimization data. Note that this accuracy does not reflect a proper evaluation of the system; see Tables 5.11 and 5.12 for proper evaluations on independent test data.

	Orig. devel.	Orig. test	Reduced devel.	Reduced test
N-attach $F_{\beta=1}$	65.71	63.11	66.74	63.00
V-attach $F_{\beta=1}$	40.93	34.47	41.35	35.29
Overall accuracy	56.61	52.80	57.55	**52.94**

Table 5.11: Results of PP attachment disambiguation experiments with SVM using default settings. The table shows F-scores for noun and verb attachments as well as overall accuracy. See the text for an explanation of the distinction between original and reduced corpus versions.

not offer any explanation for the poor performance of the SVM on the PP attachment disambiguation task.

5.13 Discussion

The idea behind the approach to disambiguation taken in this work is that the probability that a PP attaches to a verb (noun) in an ambiguous context is related to the question of whether the verb (noun) occurs *unambiguously* modified by this PP in the language. This idea is related to the ideas underlying other semi-unsupervised work, such as that by Hindle and Rooth (1993) and Ratnaparkhi (1998).

The present system (using either a memory-based learner or a MaxEnt model) has a performance which is very similar to that of Hindle and Rooth (1993)'s disambiguator. However, by taking advantage of existing supervised machine learning mechanisms and providing these mechanisms with training material in an unsupervised fashion, the method arrives at this performance level in a much simpler way, while still requiring a minimum of manual effort. Also, as shown in the previous sections, the method described here has the additional advantage that one learning algorithm, such as a memory-based learner, can easily be replaced by alternative vector-based supervised learning algorithms. Although none of the other algorithms that were tested improved the results, trying out alternative learning algorithms in the future will only require a small amount of work.

	Orig. devel.	Orig. test	Reduced devel.	Reduced test
N-attach $F_{\beta=1}$	65.71	65.91	66.74	65.91
V-attach $F_{\beta=1}$	40.93	25.04	41.35	26.18
Overall accuracy	56.61	53.13	57.55	**53.35**

Table 5.12: Results of PP attachment disambiguation experiments with SVM using optimized settings. The table shows F-scores for noun and verb attachments as well as overall accuracy. See the text for an explanation of the distinction between original and reduced corpus versions.

On the other hand, the semi-unsupervised system presented by Ratnaparkhi (1998) performs considerably better than that developed by Hindle and Rooth (1993), with an accuracy of 81.9%. A later unsupervised system for English, presented by Pantel and Lin (2000), reaches an even higher accuracy of 84.3%, which is actually as good as or better than most supervised systems.

It turns out, however, that the bulk of these performance improvements can be explained by the fact that the English preposition *of*, which is relatively frequent, virtually always attaches to the preceding noun. Unlike Hindle and Rooth (1993), the later authors take advantage of this fact and employ a special rule that always selects noun attachment for *of*. Since *of* occurs as the preposition in as many as 30% of their test cases, this has a huge impact on their results—in fact, Ratnaparkhi (1998) reports that his performance on cases where $p \neq$ *of* is only 74.6%.

Since Norwegian does not have any prepositions with (near-)unambiguous attachment, we cannot make use of such rules for this language. However, for illustrative purposes I have run an experiment in which I simulate the English state of affairs in the following way: I randomly select 30% of the Norwegian test cases (corresponding to the 30% proportion of *of* in the English test data) and pretend that they are correctly attached, whether or not they are actually correctly handled by my system. In this experiment, the accuracy of the system (using TiMBL) reaches 83.1%, i.e., an accuracy which is actually higher than that of Ratnaparkhi's system, although it is still about 1 per cent below the performance of Pantel and Lin (2000).

Of course, this is not a valid result for Norwegian, since the experiment involves distorting the data to make them look more like English. Nevertheless, it supports the idea that many of the improvements obtained by the later English systems are caused by the properties of English with respect to the preposition *of*. By the same token, it indicates that these alternative disambiguation methods might offer less of an advantage for Norwegian disambiguation than the differences in accuracy scores seem to suggest.

With regard to Ratnaparkhi's system, it should also be mentioned that it was trained and tested exclusively on the *Wall Street Journal*, which is a rather specialized

newspaper. The Norwegian system has been tested on *Bergens Tidende*, which is a non-specialized daily newspaper, and it has been trained on a wide variety of text types. I believe that the greater heterogeneity in my training data might yield a system that can handle a large variety of text types in a more robust way.

An interesting aspect of the Pantel and Lin (2000) system is the fact that the authors utilize the notion of *contextually similar words*. They obtain measures of semantic similarity between nouns and verbs from a corpus-derived thesaurus. Additionally, they use a collocation database that provides sets of words that occur in the same syntactic dependency relationship (e.g., the set of nouns occurring as the object of the verb *eat*, or the set of verbs for which *salad* occurs as an object).

The intersection of a set of similar words taken from the thesaurus and a set of words that occur in the same dependency relationship constitutes a set of contextually similar words, i.e., words that have similar meaning in a certain context. Using sets of contextually similar words instead of individual verbs and nouns reduces the problem of data sparseness, thereby improving the performance of the model.

Considering the success of Pantel and Lin (2000)'s model, implementing a similar system for Norwegian seems like a promising direction for future work. The required information about collocations and semantic similarity is currently not available, but work by Velldal (2003) on the clustering of Norwegian nouns seems to provide a good starting point for extracting such information. However, it should be noted that, although the use of semantic similarity seems intuitively on the right track and consistently improved the results in all of Pantel and Lin (2000)'s experiments, the improvements were in fact not great enough for the authors to claim with certainty that this information improves performance in general.

5.13.1 The performance of human annotators

In order to establish an upper bound on what PP attachment disambiguators can be expected to achieve, Ratnaparkhi et al. (1994) performed two trials in which three treebanking experts annotated 300 randomly selected examples from the Penn Treebank (Marcus, Santorini, and Marcinkiewicz, 1993). In the first trial, they were only given the so-called "head words", i.e., the main verb, the noun, the preposition, and the noun functioning as prepositional complement. Their average performance, measured against the markup in the treebank, was 88.2%. In the second trial, they were given the entire sentence, and the average performance increased to 93.2%. These figures have established themselves in the literature as the upper bounds for PP attachment disambiguators—in particular the 88.2% figure, since that is the one obtained in the case where only the head words are known, which is typically the situation with automatic disambiguators.

These figures do raise a number of questions, however. First of all, one might wonder why Ratnaparkhi et al.'s treebanking experts only reached an average accuracy of 93.2%—after all, it means that in almost one out of every ten cases, they made a different decision from that found in the Penn Treebank. Although correct disambiguation of some cases might require a context that is wider than a single sentence, at least in the present project these cases turned out to be quite rare. Moreover, if the problem was that the sentence did not provide sufficient information for disambiguation, we would still expect the annotators to get even these cases right half of the time on average.

Hence, we need to look for additional explanations for the high number of mistakes. First, can we be certain that it is the treebank, and not the experts, that is correct in all of these cases? Furthermore, could the "errors" be caused by the use of different guidelines, or by the fact that the intuitions of the annotators did not correspond to the guidelines used in the treebank? And although machine learning systems are generally compared to the lower performance level of 88.2%, which was obtained when only the four head words were considered, these concerns apply equally well to that figure.

A partial answer to these questions is found in a less-cited figure from Ratnaparkhi et al., which measures the accuracy of the treebank against those 274 cases in which all the treebanking experts agree. The accuracy of the treebank is 95.7%, meaning that there is a fair amount of cases in which the treebank annotation does not agree with the intuitions of the experts. Hence, if the treebank were annotated in accordance with those intuitions, we would expect the performance of the experts to be considerably higher than 93.2% and 88.2%, respectively. In fact, Ratnaparkhi et al. report that, on this subset, the experts had a 92.5% score when considering only the head words.

Thus, it could be argued that 92.5% is a better measure of human performance when they are given only the head words. An accuracy of 88.2% might still be the most appropriate score to use when evaluating machine learning methods on the entire set of 300 examples, but even then the relationship between the guidelines used for treebank annotation, the intuitions of the experts, and the decisions made by the algorithm in unclear cases should be taken into account.

On a slightly different note, it should be mentioned that Ratnaparkhi et al. carried out yet another annotation trial with three non-experts on 200 examples from a different corpus. Annotation was done based only on the four head words, and in this case, the average performance was 77.3%. Given the small number of annotators, the fact that the corpora were different, and the complications associated with the use of different guidelines, it is not clear how much weight to put on the difference in performance between the experts and the non-experts. Nevertheless, given the relatively large dif-

ference between 88.3% and 77.3%, we might still ask whether the experts were using their treebanking experience to obtain a better match to the treebank annotation.

This further leads to the question of whether we want a PP attachment disambiguator to mimic the behaviour of treebanking experts or that of native speakers who are not trained in treebank annotation. Although the focus on expert performance that is found in the machine learning literature seems to imply the first alternative, it is not clear whether this is the right answer, particularly if we want our disambiguator to be part of some kind of language understanding system.

Thus, if the difference in performance is due to the treebanking experts having clearer intuitions about correct attachment, and being better able to use appropriate techniques such as topicalization and substitution to clarify them, mimicking their behaviour would probably be a desirable goal. On the other hand, the difference might be due to the experts following explicit guidelines that aim at giving a consistent treatment of certain constructions, but that might in some cases result in questionable attachments (cf. my comments on Hindle and Rooth's treatment of light verb constructions in section 5.5.2). In that case, the decisions made by the experts do not necessarily constitute a more appropriate goal for the automatic disambiguator than those made by the non-experts.

Finally, it is worth noting that Mitchell and Gaizauskas (2002) report preliminary results from annotation trials on the same Penn Treebank data that were used by Brill and Resnik (1994) to test their Transformation-Based Learning approach. On these data, three language experts obtained a score of 75.7%. Like Ratnaparkhi et al.'s non-experts, these annotators performed far worse than various machine learning techniques that have been tested on the Penn Treebank. I see this as further support against the view that machine learning systems need to reach the 88.2% score of Ratnaparkhi et al.'s experts in order to exhibit a "human-level" performance.

Considering the abundance of cases in which it is not at all obvious how to make a decision, the performance level of human annotators will necessarily depend on the choice of guidelines for difficult cases, and such a choice is likely to have consequences for the degree to which different machine learning algorithms match human performance. Furthermore, as mentioned above, we need to decide whether we want human-level performance to be defined by the performance of trained treebanking experts or by that of untrained native speakers.

5.14 Further extensions

This chapter focuses on disambiguation of PP attachment in a certain ambiguous context, viz. sequences consisting of a verb, a (base) NP, and a PP. This is the type of context that has been the focus of most earlier work on machine learning for PP

attachment disambiguation. It is also the type of context that presents the "purest" environment for evaluating different disambiguation methods that are based on lexical preference, since lexical preference seems to be the dominating factor in determining PP attachment in this type of context.

On the other hand, this disambiguation mechanism does not in itself constitute the kind of full-blown, working system for PP attachment that is needed in a higher-level NLP application such as an anaphora resolution system. Such a system needs to incorporate the results of the disambiguation task, but it also needs to handle other types of contexts:

- PPs with syntactically unambiguous attachment, such as those occurring at the beginning of sentences or directly following the verb, and

- Consecutive PPs.

The first type is not particularly challenging, since it involves cases of unambiguous attachment which can be easily handled. The second type, however, is more problematic. As an example of this kind of context, consider (18-a) and (18-b):

(18) a. De sitter i stua.
they sit in living-room-the
"They are sitting in the living room."

b. De sitter ved spisebordet i stua.
they sit at dining-table-the in living-room-the
"They are sitting at the dining table in the living room."

c. De sitter ved motorsykkelen i stua.
they sit at motor-cycle-the in living-room-the
"They are sitting by the motor cycle in the living room."

The sentence (18-a) shows that the PP *i stua* readily attaches to the verb *sitter*. In (18-b), however, a reader is very unlikely to interpret *i stua* as a modifier of *sitter*—the PP will most certainly be interpreted as modifying *spisebordet*. Even in a sentence like (18-c), noun attachment is much more likely than verb attachment, even though the noun denotes something that is very rarely found in a living room.

Note that since Norwegian verbs are not marked for aspect, it is possible to interpret *sitter* in (18) with a habitual aspect. With this interpretation, (18-b) could be translated into *They sit at the dining table in the living room.* Given a habitual reading of *sitter*, one interpretation of the sentence is that whenever *they* are sitting somewhere in the living room, it is always at the dining table.

In such an interpretation, we would actually have verb attachment. However, this interpretation would require strong support from a wider context, and in spoken language it would require prosodic expression in the form of contrastive stress

on *spisebordet* and *stua*[10]. For this reason, I consider these cases to be syntactically ambiguous but pragmatically unambiguous. Furthermore, since the present system considers neither discourse context beyond the sentence level nor evidence from spoken language, the last PPs in sentences such as (18-b) and (18-c) will be considered unambiguously attached to the noun. Note that this will also happen, and give the wrong analysis, in cases where the last PP indicates that the verb should be given a habitual reading, e.g., in sentences corresponding to *We dine at the dining table in our family/in this country*, since the system does not have access to information about which ADVERBIALS bring forth a habitual reading of the verb.

The lack of ambiguity in cases like (18-b) and (18-c) could be an effect of the fact that Norwegian verbs are rarely modified by more than one place adverbial. Alternatively, it could be attributed to the Right Association principle mentioned in section 5.4, which states that new constituents should be attached as low as possible in the parse tree. Whatever the correct explanation, this phenomenon shows that factors other than just lexical preference come into play in such contexts, and, in a more complete system than the one developed here, we would need to deal with those factors as well.

5.15 Conclusions

In this chapter, I have presented a system for PP attachment disambiguation in Norwegian that is based on what I call pseudo-supervised learning, in which an automatic procedure is used to obtain training data for a supervised learner. In my system, both the memory-based learner and the maximum entropy model achieved an accuracy of 75.8%, which is identical to the performance of one of the most renowned semi-unsupervised systems that has been developed for English (Hindle and Rooth, 1993). Furthermore, it obtains this performance using a simpler and more flexible approach than that employed by Hindle and Rooth. It is simpler because it makes use of existing supervised learning mechanisms rather than specially crafted statistical calculations, and it is more flexible in that different vector-based, supervised learning mechanisms can easily be plugged into the system.

The performance of the system still lags slightly behind the best semi-unsupervised disambiguators for English, but this difference can largely be explained by the fact that the preposition *of* has near-unambiguous attachment in English.

In the present context, the primary motivation for creating a PP attachment disambiguator has been to be able to determine whether an NP is embedded in another

[10]It is also possible to get verb attachment for *i stua* without a habitual reading of *sitter* by moving *i stua* closer to the verb: *De sitter i stua ved spisebordet* "They are sitting in the living room by the dining table", in which case both PPs are attached to the verb. However, I consider such constructions to be rather awkward, if not directly ungrammatical.

NP (as it will be if it is the complement of a preposition that attaches to a noun), because earlier work on anaphora resolution has found that embedded NPs are less likely to occur as antecedents of pronominal anaphors than non-embedded NPs (Lappin and Leass, 1994). In chapter 8, I will investigate to what extent this finding is transferable to data-driven anaphora resolution on Norwegian fiction texts. Regardless of this question, however, resolution of PP attachment is an important function which is required by a variety of higher-level NLP tasks. Hence, the creation of a PP attachment disambiguator constitutes a valuable addition to the existing set of NLP tools for Norwegian.

- We have seen a decrease in public support of 8-9 percent this year.
- Vi har hatt en nedgang i publikumsoppslutningen på 8-9 prosent det siste året .

When I agreed to a new period at the meeting in 1993,
Da jeg på kretstinget i 1993 sa ja til en ny periode ,

In the middle of the month there will be two downhill races in Cortina.
I midten av måneden skal det kjøres to utforrenn i Cortina .

This is the first time this boy from Gjovik has advanced to the podium in a single race in the World Cup.
Det er første gang Gjøvik-gutten har tatt steget opp+på seierspallen i et enkeltrenn i verdenscupen .

So far, coming third in the combination in Chamonix just before the Olympics last year has been his best achievement in the World Cup.
Fra+før var tredjeplassen i kombinasjonen i Chamonix like før OL i_fjor hans beste verdenscupplassering .

said Harald Christian Strand Nilsen til NTB etter pressekonferansen.
sa Harald Christian Strand Nilsen til NTB etter pressekonferansen .

when I started the second run today,
da jeg startet andreomgangen i dag ,

Now the two top clubs will meet in Parma tomorrow.
Nå møtes de to toppklubbene i Parma i_morgen .

but have a delayed match (against Milan).
men har en kamp til gode (mot Milan) .

Figure 5.1: The beginning of the development corpus used for PP attachment disambiguation, showing the HTML format that was used for this task, as discussed in section 3.2.

Chapter 6

Finding Animate Nouns

6.1 Introduction

As mentioned in the introductory chapter, a crucial requirement for high-quality anaphora resolution (AR) is to have access to information about whether a certain noun denotes an animate or an inanimate entity, because the choice of third-person pronoun for a pronominal anaphor tends to be constrained by the animacy of its antecedent. In English, for example, the pronouns *he* and *she* usually refer to humans or to other kinds of personified entities, such as pets, mythical figures (e.g., *Zeus* or *Odin*), or fictional creatures (e.g., *King Kong* or *Gollum*). The pronoun *it*, on the other hand, is reserved for inanimate entities, while *they* can refer to both animates and inanimates.

The situation in Norwegian *bokmål* is parallel to that of English, with the pronouns *han* "he", *hun* "she", *den/det* "it", and *de* "de" being used in the same way as their English counterparts. Norwegian *nynorsk*, on the other hand, is more like Dutch, for example, in that the pronouns *han* "he" and *ho* "she", which are used to refer to animates of masculine and feminine *natural* gender (i.e., sex), are also used to refer to inanimate nouns of masculine and feminine *grammatical* gender, respectively. In other words, for inanimate nouns, the choice of pronoun depends on grammatical gender, while for animate nouns natural gender is the decisive factor. Hence, in both systems, information about animacy is crucial for selecting the correct pronominal anaphor.

As a consequence of this, many AR systems have used information about animacy in one way or another. Since most of the AR work has been carried out on English, the primary source of information about animacy has been WordNet (Fellbaum, 1998). Although the beginnings of Norwegian WordNet-like initiatives are described by Nygaard (2006) as well as Dyvik (2002, 2003), Lyse (2003), and Thunes (2003)), no comprehensive word net currently exists for Norwegian, meaning that this kind

of information is not readily available. There are, however, some sources of animacy information created by previous research. Jónsdóttir (2003) and Øvrelid (2003) have extracted lists of animate nouns from a preliminary version of the SIMPLE semantic lexicon for Norwegian[1], and Holen (2006) gathered lists of geographic locations from the web pages of the Norwegian Language Council[2] in order to detect nouns denoting persons that are born or living in these locations (such as *engelskmann* "Englishman" or *østfolding* "person born or living in the *Østfold* area"). Nygaard (2006), mentioned above, describes a method for automatic creation of a Norwegian word net from a dictionary, and he has constructed a database from which it is possible to extract nouns that are descendants of the *person* node in his word net.

These previous efforts have provided valuable noun collections containing a fair amount of animate nouns. For example, Holen merged her own noun lists with those collected by Jónsdóttir (2003) and Øvrelid (2003) and obtained a total of 2002 nouns. Nevertheless, a large proportion of animate nouns encountered in new texts will inevitably not be recognized as such (especially considering the fact that Norwegian is a compounding language). Hence, any new technique which is able to harvest large amounts of animate nouns should be able to provide valuable information for a Norwegian anaphora resolution system.

In this chapter, I describe two closely related methods for obtaining animacy information for large numbers of Norwegian nouns. This animacy information will be used by the anaphora resolution system described in chapter 8. The methods are based on automatic extraction from the World Wide Web and therefore have a higher recall than earlier methods, meaning that many previously unrecognized animate nouns should now be recognized as such. On the other hand, being based on automatic extraction from the noisy material that constitutes the Web, the precision of these methods is typically somewhat lower than that of previously used methods, which depend on the use of manually crafted lexical resources.

6.2 Earlier work

The methods described here are certainly not the first attempt at obtaining animacy information for anaphora resolution. Hale and Charniak (1998) use a statistical method in which they run an anaphora resolution system (Hobbs (1978)'s "naïve" approach; cf. section 7.2.3) on a text, and then count the number of times each NP has been linked with a gender-marked pronoun. They report a 68.15% accuracy on proper names (no results were given for common nouns). Denber (1998) uses WordNet to

[1] Adapted from the Danish SIMPLE lexicon; see `http://cst.dk/simple/index.html`.
[2] `http://www.sprakradet.no`.

identify animate nouns by checking whether the word *creature* is mentioned anywhere in the hypernym hierarchy or the word. Cardie and Wagstaff (1999) and Hoste (2005) also use WordNet to obtain information about animacy.

Ng and Cardie (2002b) include a machine learning feature which indicates whether the anaphor and the antecedent (candidate) agree with respect to animacy. Although they do not state where this information is taken from, it is probably extracted from WordNet, since WordNet is used to provide various other types of semantic information to the system.

Evans and Orasan (2000) combine information from WordNet, a first-name gazetteer, and a small set of heuristic rules in order to identify animate entities in English text, thereby improving the performance of a modified version of the anaphora resolution system presented in Mitkov (1998). Orasan and Evans (2001) use a combination of WordNet and memory-based learning on the same task with even better results, achieving an accuracy of 97%.

Although the aim of the present system is to extract words belonging to a particular semantic category (i.e., animate nouns) rather than to identify lexical relations between words, it was nevertheless inspired by previous work which uses word patterns to extract such relations. This work includes Hearst (1992), Berland and Charniak (1999), Caraballo (1999), and Meyer (2001).

The idea of using word patterns to mine the World Wide Web, rather than a particular corpus, to obtain information for anaphora resolution was introduced by Markert and Nissim (2005). Markert and Nissim use such patterns to check for the most likely antecedent among a set of candidates, by instantiating each pattern with the anaphor and each of the antecedent candidates in turn, searching the Web with these instantiations, and counting the number of hits. The candidate which yields the highest number of hits is taken to be the most likely antecedent. Similarly, Yang and Su (2007) present a technique for automatically acquiring patterns that are suitable for extracting semantic relations from the Web, and these relations are used to improve the performance of a coreference resolution system.

My own work is inspired by Markert and Nissim's use of word patterns to search the Web. Unlike the patterns used by these authors, however, my word patterns are not meant to represent particular semantic relations between an anaphor and its antecedent. Rather, they are geared towards finding antecedent candidates of a particular type, i.e., animate nouns.

One of the approaches I have implemented nevertheless shows similarities to Markert and Nissim's method in that the patterns are instantiated with each antecedent candidate in turn, and the number of hits are counted. In the other approach, however, the patterns are not instantiated with particular candidates. Rather, large numbers of hits are collected for each pattern, and potentially animate words are extracted from

the snippets returned by the search engine. Each of these methods will be described in more detail in the following sections.

6.3 Mining the Web with search patterns

Corpora like the Oslo Corpus of Tagged Norwegian Texts and the Bergen News Corpus are extremely valuable resources for tasks that require linguistic preprocessing of texts (e.g., lemmatization, grammatical tagging, and shallow dependency parsing). On the other hand, when the aim is to acquire as many examples of a certain construction type as possible, and no preprocessing is needed to find those constructions, no available corpus can measure up to the World Wide Web because of its exceptional size. Although the number of Norwegian documents on the Web is small in comparison to the number of English documents, it still constitutes a huge text collection.

The technique I use for harvesting animate nouns is to run a number of queries against the Google search engine[3]. Google was selected partly because of its current position as the most widely known and used search engine in the world. More importantly, however, it offers an API for doing automatic queries, i.e., queries performed by a computer program instead of by a person using the ordinary search interface. At the time these experiments were carried out, Google seemed to be the only search engine to offer such a functionality, although at the time of writing, other search engines such as Yahoo! and MSN offer similar services (Rømcke (2008) is an example of current work that utilizes query APIs from other search engines, specifically MSN). Running automated queries on the ordinary search interface is explicitly prohibited both by Google and by most other search engines, either by returning some kind of "permission denied" response to user agents that do not identify themselves as an accepted web client (such as one of the major web browsers), or implicitly by blocking IP numbers that seem to perform automated queries[4].

The majority of the queries submitted to the search engine are specifically designed to elicit animate nouns, although some are more general and designed to extract either animate or inanimate nouns depending on certain variables. Queries belonging to the former type are expected to have a high precision but a relatively low recall (due to their high specificity), while those of the latter type are expected to be high recall but low precision. Experiments with both kinds of query have been carried out with the anaphora resolution systems presented in chapter 8.

[3] `http://www.google.com`

[4] This is the case, for instance, with Kvasir (`http://www.kvasir.no`), a search engine which is specifically geared towards Norwegian queries and which therefore might have been especially valuable as a source of information about Norwegian language had it not been for the fact that it blocks attempts at automatic harvesting of search results.

6.3.1 The general idea

I describe two mining approaches, one offline and one online method. Both methods rely on the assumption that words which typically occur as the SUBJECT COMPLEMENT of a clearly animate SUBJECT tend to be animate, while those that occur as the SUBJECT COMPLEMENT of an inanimate SUBJECT are typically inanimate. Since Norwegian singular third-person pronouns are clearly marked for animacy, they are suitable for use as SUBJECTs in this context. Hence, I use *han* and *hun* as animate SUBJECTs and *den* as an inanimate SUBJECT.

I insert each of these SUBJECTs into a selection of syntactic constructions that typically take a SUBJECT COMPLEMENT. With the offline method, the set of constructions is then sent off to the Google API, after which the search results are grammatically tagged and the SUBJECT COMPLEMENTs are extracted. With the online approach, a particular noun is inserted as the SUBJECT COMPLEMENT of the same constructions before the set is sent off to Google. With both methods, the results with animate and inanimate SUBJECTs are compared, but dealt with in slightly different ways.

6.3.2 Search patterns

The search patterns, or queries, that are used are generated from the *templates* shown in Table 6.1, the *fillers* in Table 6.2, and one of the pronouns *han* "he", *hun* "she", or *den* "it (masc./fem.)"[5]. The filler constructions were selected based on my own intuitions about what would be useful patterns for extracting animate nouns.

The fillers numbered 1-6 in Table 6.2 do not contain any lexemes that are particularly associated with animate SUBJECTs, and they are therefore considered to be high recall/low precision. The remaining fillers contain a verb, SUBJECT COMPLEMENT, or OBJECT that typically occurs with an animate SUBJECT. Hence, these fillers are assumed to be low recall/high precision.

A query is constructed by inserting a filler from Table 6.2 into a slot in one of the templates found in Table 6.1. The slots are marked by tense indicators surrounded by hash symbols (#PRES#, #PAST#, #PERF#, or #INF#). The tense indicators show the required tense form of the filler: present tense, past tense, perfect participle, or infinitive, respectively (Table 6.2 only shows the infinitive form of each filler).

To complete the query, a SUBJECT pronoun—either *han*, *hun*, or *den*—is added to the construction. For some of the fillers in Table 6.2, this is probably not strictly necessary in order to obtain animate nouns, because the fillers contain predicates that in themselves select animate complements, such as *jobbe som* "work as" and *være ansatt som* "be employed as (a)". However, there are others, such as *være* "be", *bli* "become", and *fungere som* "function as", which may very well occur with

[5] *det* "it (neut.)" was excluded due to its frequent occurrence as a pleonastic pronoun.

Template	Translation
#PRES#	#PRES#
#PAST#	#PAST#
har #PERF#	has #PERF#
hadde #PERF#	had #PERF#
vil #INF#	{will/wants to} #INF#
skal #INF#	{will/is going to} #INF#
kan #INF#	{can/may} #INF#
vil ha #PERF#	will have #PERF#
skal ha #PERF#	{is said to} have #PERF#
kan ha #PERF#	may have #PERF#
ville ha #PERF#	would have #PERF#
skulle ha #PERF#	should have #PERF#
kunne ha #PERF#	could have #PERF#
har villet #INF#	has wanted to #INF#
har skullet #INF#	has been said to #INF#
har kunnet #INF#	has {been able to/had the opportunity to} #INF#
hadde villet #INF#	had wanted to #INF#
hadde skullet #INF#	had been going to #INF#
hadde kunnet #INF#	had {been able to/had the opportunity to} #INF#
skal ha villet #INF#	is said to have wanted to #INF#
skal ha kunnet #INF#	is said to {have been able to/have the opportunity to} #INF#
skulle ha villet #INF#	was said to have wanted to #INF#
skulle ha kunnet #INF#	was said to {have been able to/have the opportunity to} #INF#
kan ha villet #INF#	may have wanted to #INF#
kunne ha villet #INF#	might have wanted to #INF#

Table 6.1: Query templates used to harvest Google for animate nouns. Alternative translations are enclosed in braces and separated by slashes.

inanimate complements. For this latter group of fillers, we need to control the animacy of the SUBJECT in order to maximize the likelihood of finding animate nouns, and for consistency and simplicity all fillers are treated this way.

The canonical SUBJECT position in Norwegian is preverbal, but (as in English) the SUBJECT is postverbal in yes/no questions. Furthermore, since Norwegian is a V2 language, when some other constituent is found before the finite verb, the SUBJECT is demoted to the position immediately following the finite verb. Thus, the set of generated queries includes both *hun er* "she is" and *er hun* "is she", and both *han har vært en* "he has been a" and *har han vært en* "has he been a".

The following list shows some examples of generated queries:

- *hun har vært en* "she has been a"
- *han har rollen som* "he plays the role of"
- *hun er fungerende* "she is acting"

Filler no.	Filler in the infinitive form	Translation
1	være	be
2	være en	be a (masc./fem.)
3	være et	be a (neut.)
4	bli	become
5	bli en	become a (masc./fem.)
6	bli et	become a (neut.)
7	jobbe som	work as (a)
8	arbeide som	work as (a)
9	være ansatt som	be employed as (a)
10	være tilsatt som	be employed as (a)
11	være ansatt i stillingen som	be employed in the position of
12	være tilsatt i stillingen som	be employed in the position of
13	være valgt som	be elected as
14	være valgt til	be elected as
15	inneha stillingen som	hold the position of
16	ha stillingen som	hold the position of
17	ha rollen som	play the role of
18	fungere som	function as
19	opptre som	act as
20	være fungerende	be be acting
21	bli fungerende	become acting

Table 6.2: Query template fillers used to harvest Google for animate nouns.

Note that the English patterns corresponding to filler patterns 1-3 and 4-6 in Table 6.2 would be indistinguishable. Unlike English, Norwegian often omits the determiner in SUBJECT COMPLEMENTs consisting of a generic noun phrase, which is what the first pattern is intended to capture. These so-called *bare indefinites* also occur obligatorily in other cases where a determiner may be found in the English equivalent, such as in fillers 7-10 in Table 6.2.

Borthen (2003, p. 160) proposes the following criteria for bare indefinites in Norwegian:

"1. A bare indefinite can occur in Norwegian if it is
a) selected as a complement by a predicate and together with this predicate (and possibly other selected elements) designates a *conventional situation type*, and
b) can be seen as a reasonable candidate for being part of a *multi word lexical entry* together with this predicate (and possibly other selected elements).

2. A *conventional situation type* is a property, state, or activity that occurs frequently or standardly in a given contextual frame (e.g. in the macro social frame) and has particular importance or relevance in this frame as a recurring property-, state-, or activity type.

3. A *multi word lexical entry* is a lexical entry that in addition to the lexical item itself specifies one or more words that this item co-occurs with (i.e. selects). The multi word lexical entry constitutes a semantic and phonological unit."

Borthen also points out (p. 161) that bare singulars are "promoted in generic sentences that generalize over situation types".

An example of a bare singular occurring as a SUBJECT COMPLEMENT is given in (1-a). The NP *mann* designates the property of being a man. There is, however, often an alternative form with the same meaning that includes the determiner, as in (1-b).

(1) a. Han er mann.
he is man
"He is a man."

b. Han er en mann.
he is a man
"He is a man."

In other cases, the determiner is obligatory, as shown by the contrast between (2-a) and (2-b).

(2) a. *Han er geni.
he is genius
*"He is genius"

b. Han er et geni.
he is a genius
"He is a genius."

Furthermore, bare indefinites are not easily modified (Borthen, 2003, p. 161), as illustrated in (3).

(3) a. *Han er snill mann.
he is kind man
*"He is kind man"

b. Han er en snill mann.
he is a kind man
"He is a kind man."

Finally, since the determiner agrees in gender with the noun, all three of the pattern variants 1-3 and 4-6 in Table 6.2 are required.

6.3.3 Mining procedure

The queries generated from the patterns and fillers in Tables 6.1 and 6.2 are sent to the Google search engine using the Google SOAP Search API[6], which makes the entire

[6] `http://www.google.com/apis/`

document database available for remote applications. Applications communicate with the service through the SOAP protocol[7], which uses XML for transmitting messages between computers. Given a particular query, the service returns a number of hits in the Google database. The returned information includes an estimate of the total number of matching documents, which is used in the online data mining approach described here. It also includes a so-called "snippet", which is a small excerpt from the matching document which includes the search query terms. This snippet forms the basis for the noun extraction which is part of the offline approach.

The Google SOAP API offers an automatic filtering option which performs two functions: firstly, it removes near-duplicate documents from the returned hits, and, secondly, it limits the number of documents returned from a single host to two. I have collected search results both with and without the filtering option, and in the filtering condition I have also made sure that there are indeed no duplicate snippets in the database (by imposing a unique key on the relevant column in the database table). It turns out that, in the unfiltered condition, large amounts of duplicates are returned by Google when it runs out of results to return, a behaviour which should probably be classified as a bug on Google's behalf. Note that although the number of noun *types* (i.e., the number of different nouns) extracted from the filtered and unfiltered conditions are not that different (cf. Table 6.4 on page 140), the token frequencies, which are used in the animacy computations, naturally exhibit a much larger difference.

In any case, for the purposes of anaphora resolution, I have found that using the filtered results works best, and the results reported in chapter 8 are obtained using the filtered version.

6.4 Approach 1: Offline queries with uninstantiated patterns

In the offline approach, queries are constructed as described in section 6.3.2 and submitted to Google as detailed in section 6.3.3. Finally, nouns are extracted from the resulting snippets as described in the following section.

When each of the 25 templates in Table 6.1 are combined with each of the pronouns *han*, *hun*, and *den* in both preverbal and postverbal positions, we obtain a set of 150 patterns. Combining each of these patterns with each of the 21 fillers shown in Table 6.2 yields a total of 3150 queries (as explained in section 6.4.2; however, many of these queries do not produce any search results from the search engine).

[7] See `http://www.w3.org/TR/2003/REC-soap12-part0-20030624/`

6.4.1 Snippet analysis and noun extraction

Each of the snippets obtained by the mining procedure described in section 6.3.3 are analyzed morphologically and syntactically by the Oslo-Bergen tagger (cf. section 3.3). The analyzed snippet is then matched against the query string (composed by a template and a filler, as described in section 6.3.2) followed by an NP. Since the tagger does not analyze the text into phrases, the NPs are instead identified using the following regular expression:

(@DET> | @ADV> | @ADJ> | @SUBST>)* HEAD-NOUN

@DET>, @ADV>, @ADJ>, and @SUBST> are syntactic tags used by the Oslo-Bergen tagger and represent the following premodifier categories: determiner, adverb, adjective, and noun, respectively. HEAD-NOUN is defined as a noun which is not tagged syntactically as a determiner (which would be the case if it were a genitive form) or a premodifying noun (such as *glass* in *et glass vann* "a glass of water", lit. "a glass water").

Finally, the head noun is extracted and classified as animate or inanimate according to the type of SUBJECT pronoun(s) it tends to occur with. If the noun occurs more often in patterns with *han* or *hun* as SUBJECT, it is classified as animate; otherwise, it is classified as inanimate.

The following list repeats the query examples from section 6.3.2, this time also showing a few of the sentences returned by Google for each query. The part of each sentence that matches the query is italicized, while the noun that will be extracted from the sentence is shown in bold.

- *hun har vært en* "she has been a"
  - *Hun har vært en* god **kollega** og vært positiv på alle måter.
  - Hun peker videre på at selv om *hun har vært en* **person** med høy offentlig profil gir ikke det grunnlag for å behandle henne som "fritt vilt".
  - Fra disse vervene gikk veien naturlig til styret og arbeidsutvalget i Oslo Idrettskrets der *hun har vært en* driftig **nestleder** siden 2000.
- *han har rollen som* "he plays the role of"
  - *Han har rollen som* **Shaggy** i den kommende filmatiseringen av Scooby-Doo.
  - Frank driver også "Frank Scott Sudio" [sic] hvor *han har rollen som* **produsent** og låtskriver.

  - *Han har rollen som* en taus **butler**, og står i bakgrunnen og betrakter handlingen som foregår i forgrunnen.

- *hun er fungerende* "she is acting"
  - *Hun er fungerende* **seksjonssjef** for forskningssatsinger og forskerutdanning i Forskningsadministrativ avdeling ved UiO.
  - *Hun er fungerende* **avdelingsdirektør** i Petroleumsseksjonen.
  - Vedtaksmyndigheten kan ikke delegeres, men undervisningsinspektøren kan likevel fatte vedtak dersom han/*hun er fungerende* **rektor**.

6.4.2 Some practical considerations

Table 6.1 contains a total of 25 verbal constructions. Since Google queries cannot contain wildcards (at least not limited to single word positions, as would be required for the present purposes), all patterns need to be fully instantiated.

Furthermore, the queries are performed using the Google SOAP Search API. Although the availability of this API is greatly appreciated, the company has put a number of limitations on the search capabilities offered by the API, which have a severe impact on the efficiency with which search results can be harvested[8]. A personal account is needed to use the API, and a single account is limited to 1000 queries per day. Furthermore, the number of results returned for an individual query is limited to 10; in order to get the next 10 results, a new (identical) query must be performed, and so on. Finally, the maximum number of results for any particular search string is 1000.

What this means is that, for any instantiated search pattern, only the first 1000 can be retrieved, and 100 queries are required to obtain all of these 1000 results (since a maximum of 10 can be retrieved from any one query). With a limit of 1000 queries per day, only 10 search patterns can be processed in a single day (given that there are actually at least 1000 documents to be retrieved for each pattern). When the 25 templates are combined with each of the 21 fillers, and these in turn are combined with each of the three SUBJECT pronouns in both pre- and postverbal position, the total number of patterns amounts to 3150.

Under these conditions, 315 days (more than 10 months) would be required to process all patterns, *if* they all returned at least 1000 patterns. As it turns out, however, many of the patterns involving complex verbal constructions return none or only a few results, meaning that considerably less time is needed in order to process

[8]The limitations described here were effective during the summer of 2006, but may be subject to subsequent change. However, as of 5 December 2006, Google no longer issues new license keys for this service (and no alternative equivalent service is offered). Although existing keys can still be used after this date, this move may be taken as a signal that there is at least no intention on Google's part to improve the terms of this service.

all of them. It also means that, in many cases, more than 10 patterns can be processed in a day, since there is no need to run further queries for a particular pattern once we run out of results for this pattern.

Thus, instead of the 3,150,000 search results we would receive if all patterns returned at least a thousand hits, the system returns no more than 186K results when filtering is turned off, and 88K with filtering turned on. Table 6.3 and Table 6.4 list the exact numbers of search results received and the number of different nouns extracted from them, respectively. The nouns are ranked according to the difference between the number of occurrences with animate pronouns (*han* or *hun*) and the number of occurrences with inanimate pronouns (*den*), and the most highly ranked nouns are taken to be those most likely to be animate. Table 6.5 lists the ten most highly ranked words, while Appendix A provides a list of all those nouns that exhibit an animate–inanimate difference of two or more.

Unfiltered results	Filtered results
186,148	88,416

Table 6.3: Number of search results received from Google.

	Unfiltered results	Filtered results
All patterns	12,721	12,163
Low-risk patterns only	4163	4087

Table 6.4: Number of different nouns extracted from Google.

Rank	Lemma	Animate	Inanimate	Difference
1	rådgiver	264	1	263
2	konsulent	196	1	195
3	journalist	184	0	184
4	leder	184	1	183
5	prosjektleder	96	0	96
6	assistent	83	0	83
7	førsteamanuensis	81	0	81
8	professor	79	0	79
9	direktør	78	1	77
10	lege	68	1	67

Table 6.5: The ten nouns found by the offline approach that are most highly ranked on the animacy scale. Frequency of occurrence with animate and inanimate SUBJECT pronouns are provided, along with the animate–inanimate difference.

	Animate - inanimate > 0	Animate - inanimate > 1
All patterns	5571	2180
Low-risk patterns only	2602	1018

Table 6.6: Number of different nouns with an animate–inanimate difference higher than zero or higher than one.

Table 6.6 shows the number of nouns that have an animate-inanimate difference above zero as well as the number of nouns with a difference above one. Restricting the set of nouns to those with a difference above one filters out a relatively large proportion of inanimate nouns, at the expense of missing a fair amount of animate ones. Hence, for purposes where precision is important, it might be preferable to use the more restricted set (see section 6.6 for an evaluation of the precision of this set). On the other hand, for tasks in which it is more important to find as many animate nouns as possible (i.e., where recall is more important), using the less restricted set might be the best solution.

Experiments have shown that the memory-based anaphora resolution system described in chapter 8 falls into the latter category: using only low-risk patterns and only nouns with an animate–inanimate difference greater than one brings the accuracy of the AR system on the development corpus down from 74.60% to 72.29%, a difference which is significant at the 5% level ($p \leq 0.041$). Hence, grabbing as many nouns as possible seems to be more important than making sure they have a high probability of being animate.

The fact that recall seems to matter more than precision is an important result for the kind of automatic text-mining techniques described in this chapter, since such techniques typically have a higher recall and a lower precision than manually constructed resources. Although the effect of using all available nouns is only significant at the 5% level, this result nevertheless indicates that text-mining techniques might become even more valuable for anaphora resolution as the amount of available Norwegian text material grows.

6.4.3 Errors

Split compounds

According to the official orthographical conventions of Norwegian, compounds should be written as a single word, possibly containing a hyphen (e.g., *sommerskole* "summer school", *vaskemaskin* "washing machine"). In practice, however, it is not uncommon to find compounds split into two or more words (as in the English tradition). This constitutes a problem for the extraction mechanism, which will be "tricked" into ex-

tracting only the first part of the compound[9]. An example is given in (4), where the word *software-konsulent* "software consultant" is written as *software konsulent*.

(4) *Han jobber som* **software** konsulent[...]
he works as software consultant
"He works as a software consultant[...]"[10]

Here, only the word *software* will be extracted—clearly a mistake, since *software* does not denote a human being, while *software-konsulent* does.

Nouns used to describe properties

Some predicative nouns describe properties of the SUBJECT rather than denote the class that the SUBJECT belongs to. This is illustrated by the following example, in which *år* "years" is the head of the SUBJECT COMPLEMENT NP and hence will be incorrectly extracted as an animate noun.

(5) *Hun var* femten **år** da hun ble giftet bort[...]
she was fifteen years when she was married away
"She was fifteen years old when she was given away in marriage."[11]

Sometimes it is not even the noun itself but a following prepositional phrase that contains the actual description:

(6) Han er en slags blanding av portrettfotograf og fotojournalist [...]
he is a kind mix of portrait-photographer and photo-journalist
"He is some kind of mix between a portrait photographer and a photo journalist."[12]

(7) Han er et eksempel på hvor viktig det enkelte menneske kan være.
he is a example on how important the individual human being can be
"He is an example of how important an individual human being can be."[13]

Intervening constituents

As described in section 6.4.1, the noun extracted by the algorithm is the head of the NP following one of the template fillers in Table 6.2. This NP is normally a SUBJECT COMPLEMENT, but sometimes an NP with a different function may occur between the filler and the real SUBJECT COMPLEMENT. For instance, the fillers *fungere som*

[9] This problem could be solved with an NP chunker tuned to this kind of mistake. The Oslo-Bergen tagger does include an NP chunker, but it is only able to handle compounds that conform to the official orthography norms. Hence, it proposes a separate NP for each part of the compound and therefore does not solve the problem.

[10] http://www.enternett.no/cgi-bin/publiser/artikkel.cgi/kommando/Meny/mal/6/kategori/Om%20Enternett

[11] http://www.vinduet.no/tekst.asp?id=250

[12] http://www.gwpa.no/sider/32/dag_thorenfeldt.html

[13] http://www.liberaleren.no/arkiv/000967.php

"function as" and *opptre som* "act as" may be immediately followed by the noun *regel* "rule". However, this is not a SUBJECT COMPLEMENT and not an animate noun, but rather the last part of the adverbial expression *som regel* "as a rule/usually". Nevertheless, it is extracted by the algorithm as an animate noun.

6.5 Approach 2: (Pseudo-)online queries with instantiated patterns

The disadvantage of the offline approach described so far is that we will only be able to gather a certain number of nouns for each pattern, especially with the limitation of 1000 results per query. Even though the number of nouns collected is fairly high (at least when high-risk, low-precision patterns are included), new text will inevitably contain a fair number of antecedent candidates which are not found in the lists. Thus, it would be very useful if we could evaluate each candidate as needed with respect to animacy rather than constructing static lists of animate and inanimate nouns.

However, with the practical limitations imposed by Google, as described in section 6.4.2, such an approach seems practically infeasible at first. For example, the machine learning-based anaphora resolution system described in chapter 8 evaluates a maximum of 20 antecedent candidates for each anaphor, stopping when it finds a suitable candidate. If a text contains, say, 200 anaphors, and we need to evaluate on average, say, 10 potential antecedents for each anaphor, we will need to perform 2000 animacy evaluations for this particular text. As mentioned earlier, an exhaustive combination of the different templates, fillers, pronouns, and pre- and postverbal SUBJECT positions yields a total of 3150 patterns. Querying Google with each of these 3150 patterns for each of the 2000 antecedent candidates would take $\frac{3150*2000}{1000} = 6300$ days, or approximately 17 years!

There are, however, ways to make this task more practically feasible. First of all, a large number of antecedent candidates will re-occur many times, and there is no need to evaluate them anew each time. Thus, by caching the animacy value of already evaluated candidates, we can achieve a huge reduction in the number of queries that are required. Furthermore, we can limit the templates and/or fillers to a subset of those listed in Tables 6.1 and 6.2, keeping only the most commonly found combinations.

I have performed an experiment in which evaluated candidates are cached and where only the first eight templates and the first three fillers are used. In fact, in order to simplify the experiment, I have evaluated all candidates up front and stored them in a database before feeding them into the anaphora resolution system; this is the reason why I choose to call this a (pseudo-)online approach instead of a purely online one. The templates and fillers that are used are listed in (8).

(8)
- Templates:
  - –være "be"
  - –være en "be a (masc./fem.)"
  - –være et "be a (neut.)"
  - –bli "become"
  - –bli en "become a (masc./fem.)"
  - –bli et "become a (neut.)"
  - –jobbe som "work as (a)"
  - –arbeide som "work as (a)"
- Fillers:
  - –#PRES#
  - –#PAST#
  - –har #PERF#

Templates, fillers, pronouns, and pronoun position (pre- or postverbal) are combined in the same way as described in section 6.4, and then an antecedent candidate which is to be tested is appended to the pattern. Some examples are given in (9).

(9)
a. han er en flyplass
   "he is an airport"
b. er han en flyplass
   "is he an airport"
c. hun har jobbet som gate
   "she has worked as a street"
d. hun arbeidet som drosjesjåfør
   "she worked as a taxi driver"

Note that, since the anaphora resolution system works on grammatically tagged text, it has access to information about the grammatical gender of each candidate. Thus, for those patterns that include different gender versions (i.e., *være en/et* and *bli en/et*), we only need to apply the version that corresponds to the gender of the antecedent candidate.

The fully expanded query is submitted to the Google SOAP API. Among the information returned by the SOAP call is an estimate of the number of matches found for this query in the Google database (the snippets are not used in this online approach). If the number of hits found for the queries using animate SUBJECTs is higher than the number of hits with inanimate SUBJECTs, the noun in question is judged to be animate; otherwise, it is deemed inanimate.

6.6 Evaluation

The results of the two animacy detection algorithms are used as input to the anaphora resolution system described in chapter 8, and their usefulness for anaphora resolution is evaluated in that chapter. However, the animacy information provided by the algorithms may also be useful outside the context of anaphora resolution, and it is therefore interesting to evaluate the quality of this animacy information in itself. As we will see in chapter 8, the offline approach proves to be more beneficial than the online one for anaphora resolution. Taking this to indicate that the offline approach produces the most useful animacy information, I have restricted the separate evaluation to this approach.

Before doing such an evaluation, a number of decisions have to be made with respect to what should be counted as a correctly identified noun and what should be counted as incorrect. Should we restrict the evaluation to exclusively human nouns, leaving out nouns that may also denote animals, such as *ettåring* "one-year-old"? And what about metaphorical uses—should *stjerne* "star" or *gris* "pig" be included because they can be used in a metaphorical sense about humans, although their literal sense is inanimate (in the first case) or refers to an animal rather than a human being (in the second case)?

For the present evaluation, I have decided to use a strict criterion for correctness: only nouns that can *only* denote human beings are considered correct. Thus, *ettåring*, *stjerne*, and *gris* are excluded. However, nouns that have both animate and inanimate senses are counted as correct as long as they have at least one exclusively human sense which is not derived metaphorically from an inanimate one. This is the case, for instance, with *guide*, which can be either a human guide or a guidebook and where the animate sense is not considered to be derived from the inanimate one.

This strict evaluation is meant to establish a lower bound on the precision of the offline approach; for purposes where more liberal criteria are acceptable, the effective precision will be higher.

Note that clear cases of incorrect lemmatization (such as *enhetslede* instead of *enhetsleder* "unit manager") are marked as correct, since the lemmatization does not affect the animacy of the noun and no inanimate noun with the given lemma exists. The same applies to misspellings such as *føstekeeper* for *førstekeeper* "first keeper". However, in cases where an inanimate noun with the given lemma exists, the noun is treated as incorrect even if it is likely to be the result of incorrect lemmatization of an animate noun. This is the case, for instance, for *dans*, *sang*, and *bygningsarbeid*. These lemmas are probably the result of *danser*, *sanger*, and *bygningsarbeider* being mistakenly analyzed as plural forms of the inanimate nouns *dans* "dance", *sang* "song", and *bygningsarbeid* "construction work", when they really should have been analyzed as singular forms of the animate nouns *danser* "dancer", *sanger* "singer",

and *bygningsarbeider* "construction worker". Since we have no way of proving that this is the case, however, they are nevertheless counted as errors.

Using this strict evaluation procedure, the highest-ranked nouns produced by the offline approach have been evaluated by a trained linguist specializing in Norwegian language studies[14]. The evaluation was performed on the 1018 nouns listed in Appendix A. These are nouns extracted from filtered results using low-risk patterns only, and are restricted to those that have an animate–inanimate difference of two or more. The rightmost column of the list in Appendix A shows whether the noun was classified as correct or incorrect.

The overall proportion of correct nouns on the entire list was 92.83%, while the 800 most highly ranked nouns contained 95% correct ones. The precision was evaluated separately on each block of 100 nouns from highest to lowest rank (i.e., rank 1-100, rank 101-200 etc., with the last block containing rank 901-1018), and these results are shown in Figure 6.1 (note that the vertical axis starts at 70% in order to display the variation more clearly). The figure also shows how the cumulative precision changes as the blocks are added one by one to the test set.

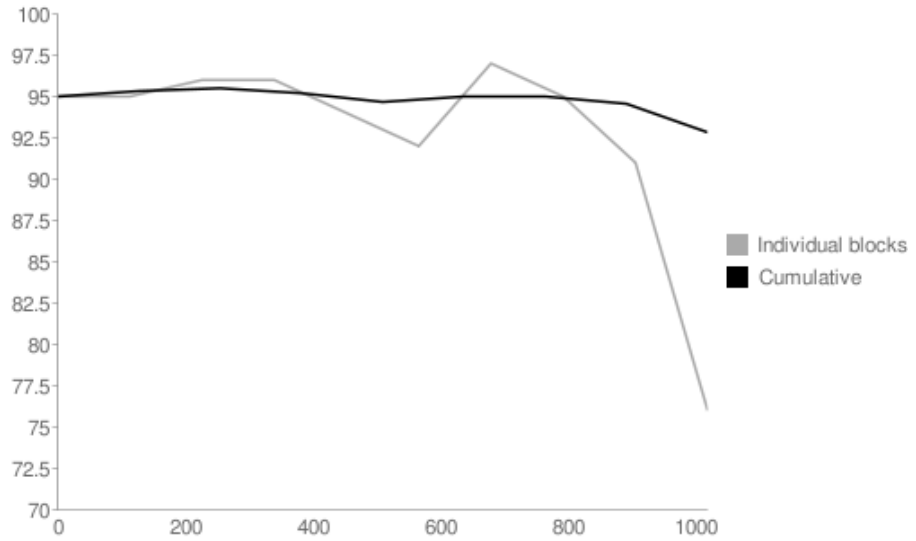

Figure 6.1: Proportion of nouns extracted by the offline approach that are correctly classified as exclusively human. The horizontal axis shows the number of nouns evaluated, while the vertical one displays precision. Note that the vertical axis starts at 70% in order to display the variation more clearly. Separate precision for each block of 100 nouns is shown, as well as the cumulative precision obtained by adding each 100-noun block to those already included.

[14]The evaluation was carried out by Åshild Søfteland.

As is evident in the figure, precision drops dramatically towards the end of the ranking scale, with the set of nouns from 900-1018 containing only 76% animate nouns. Hence, for tasks requiring high precision, it might be advisable to use only the 800-900 most highly ranked nouns. However, as pointed out in section 6.4.2, for the task of anaphora resolution, using as many nouns as possible turns out to be the best strategy despite the falling precision.

Note that the limit of 800-900 suggested here only applies to the particular list that was extracted in the present experiments, using the Google API at a specific point in time. Employing the Google API again at a later point in time or using a different search engine altogether might yield different numbers of high-precision nouns. All nouns between the ranks of 567 and 1081 have a difference between animate and inanimate occurrences of two (mostly having two animate and zero inanimate occurrences, but in two cases there are three animate occurrences and one inanimate one). Thus, there is no reason to believe that the 800-900 limit will generalize to other experiments. On the other hand, the present experiments do suggest that, in general, limiting the set of nouns to those with an animate–inanimate difference of at least three might be good if high precision is more important than high recall.

6.7 Conclusions

In this chapter, I have presented two approaches to the problem of determining whether a noun is animate or inanimate. Both approaches use the World Wide Web as their source of information and employ the Google search engine to extract relevant information from this source. The approaches differ in some important respects, however. One approach applies so-called "offline processing" in which large numbers of nouns are extracted from the snippets that Google returns. The other one, called the "online" (or "pseudo-online") approach, inserts the antecedent candidate nouns directly into the search patterns and counts the number of times they co-occur with animate and inanimate SUBJECT pronouns, respectively.

It was hypothesized that animacy information would be important for anaphora resolution, and this hypothesis has been tested as described in chapter 8. In that chapter, it is shown that this kind of information is very valuable indeed, at least for the resolution of the *han/hun* and *den* pronouns. The experiments also compare the online and offline methods and find that the offline method is by far the most valuable. In the present chapter, the offline approach has also been evaluated separately and has proven to extract animate nouns with high precision, even when a rather strict criterion for correctness is adopted.

Chapter 7

The Field of Pronominal Anaphora Resolution

7.1 Introduction

This chapter is concerned with the research field that constitutes the main focus of this thesis, i.e., the field of *pronominal anaphora resolution* (AR). The goal of the AR task as it is defined in this thesis can be summed up as follows: for any given pronoun in a text, find the closest preceding NP that the pronoun is coreferent with (i.e., its antecedent).

Consider the following example, which was also given in the introductory chapter. In (1-a), *det* "it" corefers with *Toget* "the train", in (1-b) it corefers with *reinsdyret* "the reindeer", while in (1-c) *det* does not corefer with anything.

(1) Toget traff reinsdyret fordi ...
train-the hit reindeer-the because
"The train hit the reindeer because ..."

a. det kjørte for fort.
it drove too fast
"it was driving too fast."

b. det sto i skinnegangen.
it stood in rails-the
"it was standing on the rails."

c. det var mørkt.
it was dark
"it was dark."

As humans, we are able to identify the antecedent in these examples through the use of our linguistic and world knowledge. This includes knowledge about morphology

(e.g., gender match/mismatch between an anaphor and a potential antecedent), syntax (e.g., the existence of constructions such as (1-c), which describe some aspect of the environment—it was dark/cold/wet etc.—and in which the pronoun is non-referential), semantics (e.g., whether trains or reindeer drive fast), and pragmatics (e.g., the fact that a pronoun functioning as SUBJECT is more likely to corefer with the constituent realizing the topic of a previous utterance than with other potential candidates in the utterance).

Given the need for such a variety of linguistic knowledge, it is hardly surprising that AR turns out to be a very hard task for automatic systems. In fact, Hobbs (1978) writes about Charniak (1972) that he "demonstrated rather convincingly that in order to do pronoun resolution, one had to be able to do everything else" (Hobbs, 1978, p. 212). Whether or not one subscribes to such a pessimistic view, it is nevertheless the fact that state-of-the-art AR systems have F-scores that are typically somewhere between 60 and 70—a rather modest performance level compared to the state of the art for most other NLP tasks, which often reach scores above 90.

7.1.1 Anaphora

The term *anaphora* is used to denote a number of discourse phenomena that involve *referring expressions*[1]. Referring expressions are expressions that are used to refer to entities in the mental representation—called the *discourse model* (Webber, 1978)—that a speaker or hearer has of the current discourse. These entities are called *discourse referents* (Karttunen, 1976). Expressions that refer to the same discourse referents are said to *corefer*.

Referring expressions may either evoke new entities in the discourse model, or they may access already established entities. Expressions that access established entities are called *anaphoric*, and earlier expressions which refer to the same entity are called the *antecedents* of the anaphor (Kehler, 2000). Another way to view such anaphoric expressions is that they do not have independent semantics, because their interpretation relies on the semantics of their antecedents.

Various kinds of anaphoric phenomena are recognized in the literature, and these phenomena involve various types of referring expressions. Kehler (2000) lists the following types of referring expressions: indefinite noun phrases (*a car*, *some cars*), definite noun phrases (*the car*), pronouns (*she*, *they*), demonstratives (*this*, *that*), and one-anaphora (*one* as in *I looked for a car, but I could not find one*). In this thesis I focus on pronominal anaphora, i.e., cases in which the anaphor is a pronoun.

Although most pronouns are anaphoric, and hence corefer with referring expres-

[1] In syntactic theory, the term *referring expression* is often used to denote lexical nouns only, not including pronouns. I am, however, adopting the terminology used by Kehler (2000), who does include pronouns among referring expressions.

sions that occur earlier in the discourse, this is not always the case. In some cases, the first or only coreferring expression occurs *later* in the discourse, in which case the pronoun is called *cataphoric* instead of anaphoric. Other pronouns are *deictic* and have a reference which depends on the non-linguistic context in which the discourse takes place—this is typically the case for first- and second-person pronouns. Deictic pronouns may or may not have an antecedent in the discourse.

Finally, some pronouns, such as English *it* and Norwegian *det*, may be *pleonastic* (also called *expletive*, *structural*, *dummy*, or *non-anaphoric* (Evans, 2001)), in which case they do not have any reference at all, and hence do not function as referring expressions.

In this thesis I will not deal specifically with cataphors or deictic or pleonastic pronouns. The various AR systems I describe will have to determine when a pronoun corefers with an earlier referring expression and when it does not, but in the latter case they will not be required to decide whether the pronoun is cataphoric or pleonastic. Furthermore, like most earlier systems, the AR systems in this thesis exclude first- and second-person pronouns from consideration, because their primarily deictic nature makes them less relevant for the main goal of the systems, which is to establish coreference chains of referring expressions.

7.1.2 Antecedent types

At the beginning of this chapter, it was stated that the goal is to find the closest *NP* that the pronoun corefers with, and it is indeed the case that most antecedents are NPs (including pronouns). Sometimes, however, the antecedent is a VP or a sentence, as illustrated in (2) and (3), respectively.

(2) I like to [play the drums]$_i$. [It]$_i$ is great fun.

(3) [It is raining today]$_i$. [That]$_i$ is boring.

Most earlier work on AR has been restricted to NP antecedents, although some work has been carried out on non-NP antecedents as well (Byron, 2002; Nararretta, 2004; Strube and Müller, 2003). In this thesis, I am focusing exclusively on NP antecedents, partly because the BREDT data are only annotated with NP antecedents. Furthermore, very little previous work has been carried out on Norwegian AR, and I believe that the best way forward is to develop systems for resolving anaphors with NP antecedents before trying to augment these systems with mechanisms for handling the task of finding non-NP antecedents, since the latter task seems to be a considerably harder one (at least judging from the low performance scores obtained by Strube and Müller (2003)).

7.2 Previous work on anaphora resolution

7.2.1 Knowledge-intensive vs. knowledge-poor approaches

The AR task has a long history, dating back to the time of Winograd (1972)'s SHRLDU system, which included AR as one part of its language module. However, the earliest algorithm that is still widely referred to (and one that still compares favourably with many of today's systems), is the so-called "naïve" approach presented by Hobbs (1978). This approach is purely syntax based and does not require access to semantic or pragmatic information or to world knowledge. In the same paper, however, Hobbs also presents a "non-naïve" approach that relies on having access to a certain amount of semantic information, in the form of selectional restrictions. Hobbs' two approaches may be seen as representative of the two main strands found in later work on AR, viz. knowledge-poor versus more knowledge-intensive approaches.

In the knowledge-intensive camp, work by Hobbs and colleagues on so-called "abductive inference" (inference to the best explanation) (Hobbs, Stickel, Appelt, and Martin, 1993; Hobbs, Stickel, Martin, and Edwards, 1988) was used in the approach to coreference resolution described by Kehler (2000, 2002). In a similar vein, Carter (1987) used the common-sense inference theory of Wilks (1975) as one of the pillars of his AR system.

Carbonell and Brown (1988) approach the AR problem with a combination of multiple knowledge sources, most of which require fairly sophisticated linguistic or world knowledge: sentential syntax, case-frame semantics, dialogue structure, and general world knowledge. Similarly, the AR module described by Rich and LuperFoy (1988) relies on the output of a semantic module that produces a set of discourse referents and a set of assertions about these referents, including semantic properties such as being an agent, being the result of a creative act, etc. Rich and LuperFoy combine this semantic knowledge with various kinds of information that are also typically used in knowledge-poor approaches, such as gender agreement, number agreement, and recency, as well as information about which entities are globally salient throughout a discourse.

Another example of knowledge-intensive anaphora resolution is given by Asher and Lascarides (2003), who present a theory called Segmented Discourse Representation Theory. Asher and Lascarides's theory is an extended version of the Discourse Representation Theory introduced by Kamp and Reyle (1993), which is a form of dynamic semantics. Dynamic semantics is an attempt at alleviating a problem found in static semantic theories such as those of Montague (1974) and Davidson (1980), viz. that these theories do not handle links between sentences. For instance, in the static semantic theories, a pronoun such as *he* is considered to be an independent free variable, which may be assigned to an individual which is different from any of the individuals

introduced earlier in the discourse. Dynamic semantics, on the other hand, contains a concept of contextual interpretation, where the sentence containing *he* is placed in a context which consists of sentences that occur earlier in the discourse, and these sentences may contain possible antecedents for the pronoun.

In this thesis, however, I am focusing on knowledge-poor approaches, and I will therefore restrict any further descriptions and discussions to approaches that I consider to be knowledge poor. It should be noted that—like most other computational linguistics systems—all of the approaches described here do require a certain amount of linguistic information, and there may be some disagreement as to where various researchers will want to draw the line between "knowledge poor" and "knowledge intensive". In my own work, I follow the classification outlined by Lappin (2005) and draw the line between, on the one hand, approaches that require access to semantic information such as selectional restrictions or to inference engines, and, on the other hand, those that do not require any linguistic knowledge beyond information about tokenization, lemmatization, and morphological and syntactic analysis. I do, however, include among the knowledge-poor approaches those that rely on estimates of the salience of different antecedent candidates, as explained below.

7.2.2 The major directions in knowledge-poor AR

Within the knowledge-poor tradition, an alternative to Hobbs (1978)'s purely syntactical algorithm was presented by Lappin and Leass (1994), whose AR system includes modelling the salience of possible antecedents. In a similar vein, the systems created by Mitkov (1998, 2002) depend on a number of so-called *indicators*, which are reminiscent of Lappin and Leass' salience factors.

Another influential knowledge-poor theory has been Centering Theory (Grosz, Joshi, and Weinstein, 1983, 1995; Poesio, Stevenson, Eugenio, and Hitzeman, 2004; Walker, Joshi, and Prince, 1998). Like the Lappin and Leass approach, centering also assumes a salience rating, but it includes the added constraint that there is a single item which is in focus at any given time, and that this item is the most likely to be pronominalized.

Although I do regard these approaches as knowledge poor, since they do not rely on sophisticated semantic or pragmatic information or on world knowledge, much recent recent work on AR has nevertheless been geared towards methods that demand even fewer resources. For example, Kennedy and Boguraev (1996) modified Lappin and Leass' algorithm so that it can operate on the output of a shallow dependency parser, thereby removing the need for correct deep syntactic analyses. Other examples of work that is less demanding with respect to syntactic analysis are Roland Stuckard's ROSANA system (Stuckard, 2001), which uses fragmentary output from a syntactic parser, and Breck Baldwin's CogNIAC system (Baldwin, 1997), which applies a set

of anaphora resolution rules to text which has been only part-of-speech tagged and base-NP chunked. Furthermore, much work has been focused on the use of machine learning methods for AR in recent years.

In the following sections, I will give an overview of some important work within these various directions in knowledge-poor methods for anaphora resolution. For the most part, I will restrict the overview to anaphora resolution proper and exclude work on the wider field of coreference resolution. When it comes to machine learning approaches, however, I will include some work on coreference resolution, since most machine learning-based work handles anaphora resolution and coreference resolution using the same mechanisms and statistical models.

7.2.3 Syntax-based approaches

Hobbs

The "naïve approach" to anaphora resolution described by Hobbs (1978) is based on a fairly complex set of rules for traversing the syntactic parse of a text. Hobbs' reason for calling this approach "naïve" is that it does not take into account any semantic knowledge; it is purely syntax based (as mentioned above, in the same paper, Hobbs also describes a non-naïve approach that also makes use of semantic knowledge, in the form of selectional restrictions).

Hobbs' system searches for antecedents in parsed syntactic trees, and the key to the success of the algorithm (given perfect parses of the text) lies in the particular order in which the tree is searched. Hobbs (1978, p. 315-316) describes the search procedure as follows:

"(1) Begin at the NP node immediately dominating the pronoun.

(2) Go up the tree to the first NP or S node encountered. Call this node X, and call the path used to reach it p.

(3) Traverse all branches below node X to the left of path p in a left-to-right, breadth-first fashion. Propose as the antecedent any NP node that is encountered which has an NP or X node between it and X.

(4) If node X is the highest S node in the sentence, traverse the surface parse trees of previous sentences in the text in order of recency, the most recent first; each tree is traversed in a left-to-right, breadth-first manner, and when an NP node is encountered, it is proposed as antecedent. If X is not the highest S node in the sentence, continue to step 5.

(5) From node X, go up the tree to the first NP or S node encountered. Call this new node X, and call the path traversed to reach it p.

(6) If X is an NP node and if the path p to X did not pass through the $\bar{N}$ node that X immediately dominates, propose X as the antecedent.

(7) Traverse all branches below node X to the *left* of path p in a left-to-right, breadth-first manner. Propose any NP node encountered as the antecedent.

(8) If X is an S node, traverse all branches of node X to the *right* of path p in a left-to-right, breadth-first manner, but do not go below any NP or S node encountered. Propose any NP node encountered as the antecedent.

(9) Go to step 4."

Note that Hobbs' method does not handle pleonastic pronouns, nor does it deal with reflexive or reciprocal pronouns.

Hobbs evaluated his system by hand on three different texts: a history book chapter, a novel, and a news article. Pleonastic uses of *it* were excluded from the evaluation. When evaluated on a total of 300 pronouns, his so-called naïve approach obtained an accuracy of 88.3%, (when selectional restrictions were included, the performance of the system increased to 91.7%).

Three decades later, these results are still some of the very best reported on the AR task. However, a problematic aspect of the algorithm is the fact that it requires correct, deep syntactic parses in the form of syntactic trees. Since automatic deep syntactic analysis of unrestricted text will normally contain a fair amount of errors, the performance of Hobbs' resolution module would probably be considerably lower if it were used as a component in a fully automatic system.

7.2.4 Centering Theory

Over the last two decades, one of the most important frameworks for the representation and resolution of anaphora has been Centering Theory. The theory builds upon earlier work on focusing in discourse (Grosz, 1977, 1981; Joshi and Kuhn, 1979; Joshi and Weinstein, 1981; Sidner, 1979), but was first formulated as such by Grosz et al. (1983), and later refined and extended by Grosz et al. (1995). A range of other authors have also contributed to the theory, and in the following sections I will present an overview of some important contributions to Centering Theory.

Grosz et al.

Centering Theory, as formulated by Grosz et al. (1983, 1995), is actually not in itself an anaphora resolution algorithm, although it has been used as the foundation for some of the best-known resolution algorithms since its conception. Rather, it is a theory

about the *local coherence* of discourse, i.e., the coherence between utterances within a discourse segment. The theory states that speakers choose referring expressions in such a way that they minimize the inference load placed upon the hearer. Example (4) is a slightly modified version of one of the examples provided by Grosz et al. in order to illustrate these notions:

(4) a. Terry really goofs sometimes.
b. Yesterday was a beautiful day and he was excited about trying out his new sailboat.
c. He wanted Tony to join him on a sailing expedition.
d. He called him at 6AM.
e. He was sick and furious at being woken up so early.
f. Tony was sick and furious at being woken up so early.

Although *he* in (4-e) refers to Tony, the hearer is initially misled into thinking that it refers to Terry because it has the form of a pronoun. An interpretation of the entire utterance is needed in order to determine that *he* actually refers to Tony. Example (4-f), on the other hand, is much more natural, because it avoids the use of the pronominal form. Grosz et al.'s point is that the choice of referring expression is determined by coherence considerations, and that suboptimal choices of referring expressions (such as the choice of a pronominal expression in (4-e)) increases the inference load placed upon the hearer.

Centering Theory defines a number of fundamental concepts. A *center* is said to be a semantic entity that serves to link utterances together; centers are naturally interpreted as something along the lines of Karttunen (1976)'s discourse referents, although the precise definition of this term is left up to the specific semantic theory chosen by the users of Centering Theory. Note that centers are semantic entities, not linguistic forms; rather, they must be *realized* by linguistic forms in the utterance.

Each utterance (a term which is also left open for further definition; see Poesio et al. (2004) for discussion) has one or more *forward-looking centers* Cf (i.e., all discourse entities in the utterance), but only exactly one *backward-looking center* C_b[2]. Furthermore, one of the forward-looking centers is the *preferred center* C_p. This is the center that is ranked highest according to some criterion (typically based on syntactic function) and is the one which is most likely to be coreferent with the backward-looking center of the following utterance, according to the transition hierarchy, rules, and constraints described below.

For example, (4-c) contains the forward-looking centers HE, TONY, and SAILING EXPEDITION, with HE being the preferred center because it is most likely to be coref-

[2]Walker et al. (1998) propose a weaker version of this principle, stating that each utterance has *at most* one backward-looking center.

erent with the backward-looking center of the following utterance, in this case the HE of (4-d).

Grosz et al. also introduce the notion of *transitions*, i.e., ways in which the local focus is updated when moving from one utterance to another. They define the following transition types on a sequence consisting of utterance U_{i-1} followed by U_i:

- CENTER CONTINUATION: $C_b(U_i) = C_b(U_{i-1})$ and $C_p(U_i) = C_b(U_i)$, i.e., the backward-looking center of U_i is the same as the backward-looking center of U_{i-1}, and the preferred center of U_i is the same as its backward-looking center.
- CENTER RETAINING: $C_b(U_i) = C_b(U_{i-1})$, but $C_p(U_i) \neq C_b(U_i)$, i.e., the backward-looking center of U_i is the same as the backward-looking center of U_{i-1}, but the preferred center of U_i is *not* the same as its backward-looking center.
- CENTER SHIFTING: $C_b(U_{i-1}) \neq C_b(U_i)$, i.e., the backward-looking center of U_i is *not* the same as the backward-looking center of U_{i-1}.

Finally, the main claims made by Grosz et al. may be organized into the following sets of rules and constraints (Brennan, Friedman, and Pollard, 1987):

- Rules:
  - If some element of $Cf(U_{i-1})$ is realized as a pronoun in U_i, then so is $C_b(U_i)$.
  - CONTINUATION is preferred over RETAINING which is preferred over SHIFTING
- Constraints:
  - There is precisely one C_b
  - Every element of $Cf(U_i)$ must be realized in U_i
  - $C_b(U_i)$ is the highest-ranked element of $Cf(U_{i-1})$ that is realized in U_i

Kameyama

Kameyama (1986) shows that it is necessary to account for the fact that SUBJECT pronouns tend to have SUBJECT antecedents, while non-SUBJECT pronouns typically have non-SUBJECT antecedents, i.e., a form of *grammatical parallelism*. She illustrates this point using parallel examples from Japanese and English. Her English examples are as follows:

(5) a. Who is Max waiting for?
    b. He is waiting for Fred. [C_b<SUBJ>=Max]

c. (i) He invited him to dinner. [strong preference: Max invited Fred]
(ii) ?*He was invited by Max to dinner.

(6) a. Who is waiting for Max?
b. Fred is waiting for him. [C_b<nonSUBJ>=Max]
c. (i) He invited him to dinner. [weak preference: Fred invited Max]
(ii) (?) He was invited by Max to dinner.

While Kameyama's parallelism constraint is intended to account for the different preferred interpretations in (5-c-i) and (6-c-i), the difference in *degree* of preference between these examples is attributed by Kameyama to an overall preference for SUBJECT backward-looking centers[3].

Brennan, Friedman, and Pollard

Brennan et al. (1987) argue that their ordering of the Cf list according to the grammatical ranking hierarchy, along with the preference for continuing over retaining, obviates the need for Kameyama's parallelism constraints.

As an illustration, if we were to treat Kameyama's examples in Brennan et al.'s terms, the strong preference for the given interpretation in (5-c-i) would be seen as a consequence of CONTINUATION in (5-b). On the other hand, the weakness of the preference for the given interpretation in (6-c-i) would be attributed to the occurrence of RETAINING rather than CONTINUATION in (6-b).

However, Kehler (1997) shows that the Brennan, Friedman, and Pollard (BFP) algorithm does not always mimic Kameyama's parallelism constraints, and that it yields incorrect results in some cases where a parallelism constraint would predict

[3]I have not conducted any experiments to determine whether a similar parallelism effect can be found in Norwegian. However, since the effect exists in such unrelated languages as English and Japanese, it is likely to be found in Norwegian as well, since that language is closely related to English (and, as we will see later in this chapter, syntactic parallelism does influence the performance of an automatic AR system for Norwegian).

As an informal test of the preference for parallelism in Norwegian, I showed the following sentence, copied from a cooking magazine, to some of my colleagues and asked them what *den* coreferred with:

(i) Mens kjøttet steker kutter du *bunnen*$_{OBJ}$ av *romanosalaten*$_{PCOMP}$ og vasker
while meat-the fries cuts you bottom-the of romano-salad-the and wash
den$_{OBJ}$ godt. (*Gourmega no. 3 2006*)
it well
"While the meat is frying, cut the bottom off the romano salad and wash it thoroughly."

Several people stated that they initially selected *bunnen* as the antecedent of *den*, which is likely due to their parallel syntactic functions as OBJECTs (although the decision could also be influenced by the fact that *romanosalaten* is embedded in a PP). In most cases, knowledge about cooking will kick in after a split second and make the reader realize that a) the thing that is washed should probably be used as an ingredient, not be thrown away, and b) the part of the salad that remains after the bottom is cut off is likely to be used as an ingredient, while the bottom itself is likely to be thrown away. Hence, eventually, *romanosalaten* turns out to be the most likely antecedent after all, in spite of the initial parallelism effect.

the correct result. Furthermore, Kehler shows that the BFP algorithm chooses different antecedents in sets of sentences for which humans intuitively select the same antecedent, and that the choice of antecedent for a SUBJECT pronoun may be influenced by the occurrences of other NPs later in the utterance. This conflicts with one of the basic motivations for Centering Theory, viz. the assumption that humans seem to select antecedents immediately, without waiting for the whole utterance to be processed.

Of course, one could attempt to create a dynamic version of the BFP algorithm in which an antecedent is selected early during utterance processing with the possibility of making another choice as more of the utterance is processed. However, even such a dynamic algorithm would end up making the wrong choice in the end if it relied on the constraints and rules that constitute the backbone of the BFP algorithm.

Following one of the suggestions made by Grosz et al. (1983), Brennan et al. (1987) rank the Cf list by grammatical function. The exact details of this ranking are not entirely clear, however. In Brennan et al.'s own words, they use the following order: "first the SUBJECT, OBJECT, and OBJECT2, followed by other subcategorized functions, and finally, adjuncts" (Brennan et al., 1987, page 156). This quotation is ambiguous, and it can be interpreted as giving (at least) either one of the following orderings:

(7)
- SUBJECT
- OBJECT
- OBJECT2
- other subcategorized functions
- adjuncts

(8)
- SUBJECT, OBJECT or OBJECT2
- other subcategorized functions
- adjuncts

Kehler (1997), on the other hand, states that, in Brennan et al. (1987), "the grammatical SUBJECT is ranked above all other grammatical relations (OBJECT, OBJECT2, and so forth)" (Kehler, 1997, page 468, footnote 2). This indicates the following ordering:

(9)
- SUBJECT
- all other grammatical functions

As shown by Kameyama (1986), subjecthood needs to be treated in a special way (at least as long as the ranking is based on syntactic criteria, rather than on information

structure; cf. the description of the work of Strube and Hahn below). This view is supported by other frameworks, such as MARS (Mitkov's Anaphora Resolution System; Mitkov, 2002), which often include features (or "indicators", in Mitkov's terms) that check whether a potential antecedent is a SUBJECT.

Thus, ordering (8) can probably be ruled out, since it does not give special priority to the SUBJECT. In my view, Kehler's interpretation (i.e., (9)), apart from the fact that the SUBJECT is given special priority, is not supported by the quote from Brennan et al. (1987). Hence, I conclude that ordering (7) is the one that was most likely intended by Brennan et al. It is also the one that is most compatible with the ordering given by Grosz et al. (1995, page 15):

(10)
- SUBJECT
- OBJECT(S)
- others

Beaver (2004) and Hardt (2004)

Brennan et al.'s algorithm, although not able to compete with more recent algorithms (see, e.g., the comparisons in Tetreault (2001)), is nevertheless an important milestone in the history of anaphora resolution. More recently, Beaver (2004) has presented a version of centering in terms of Optimality Theory (Prince and Smolensky, 1993) and shown that his algorithm is equivalent to that of Brennan et al. with respect to the predictions they make about anaphora resolution. Beaver proposes the following set of constraints:

- AGREE: Anaphoric expressions agree with their antecedents in terms of number and gender
- DISJOINT: Co-arguments of a predicate are disjoint
- PRO-TOP: The C_b is pronominalized
- FAM-DEF: Each definite NP is familiar
- COHERE: The C_b of the current sentence is the C_b of the previous sentence
- ALIGN: The C_b is in subject position

Hardt (2004) gives an illustration of Beaver's algorithm using the following example taken from Grosz et al. (1995), and shows that Beaver, like Grosz et al. (1995), correctly prefers reading (13-a) to (13-b) and reading (13-c) to (13-d):

(11) Susan$_i$ gave Betsy$_j$ a pet hamster.

(12) She$_i$ reminded her$_j$ that such hamsters were quite shy.

(13) a. She$_i$ asked Betsy$_j$ whether she liked the gift.
b. Susan$_i$ asked her$_j$ whether she liked the gift.
c. Betsy$_j$ told her$_i$ that she really liked the gift.
d. She$_j$ told Susan$_i$ that she really liked the gift.

Given the following example, however, Hardt shows that Beaver incorrectly predicts that the so-called sloppy reading in (16-b) is unacceptable.

(14) Ellen$_i$ was talking to Mary$_j$ and Susan$_k$ about cats.

(15) Mary$_j$ loves her$_j$ cat.

(16) a. Susan$_k$ loves her$_j$ cat too (*strict*)
b. Susan$_k$ loves her$_k$ cat too (*sloppy*)
c. Susan$_k$ loves her$_i$ cat too (*other*)

Hardt (2004) then goes on to present his own OT-based centering variety, which he calls Dynamic Centering. He removes the COHERE constraint, which he finds to be unsupported in the literature, and introduces a notion of incremental center shifting, resulting in the following definitions and constraints (Hardt, 2004, p. 4):

- "Definitions:
  - CENTER ESTABLISHER: an NP of the form NP* becomes the new CENTER.
  - CENTER: at any point in the discourse, CENTER is the most recently occurring CENTER ESTABLISHER
- Constraints:
  - PRO-TOP-INC: An utterance must contain a pronoun referring to the center.
  - ALIGN: Center establisher appears in Subject position
  - COHERE: eliminated"

With these changes, the center in (16-b) above is allowed to shift to Susan, making the sloppy reading acceptable while still ruling out the unacceptable *other* reading in (16-c). Hardt goes on to show that his changes to the model do not prevent it from yielding the correct predictions on several classic examples from the Centering literature.

Intermezzo: Subject vs. Topic

Kameyama (1986)'s parallelism constraint, discussed above, refers to SUBJECT vs. non-SUBJECT. However, it might be more appropriate to talk about TOPIC vs. non-TOPIC. The distinction between SUBJECT and TOPIC tends to be blurred in English, where TOPICs are normally expressed as SUBJECTs. In Scandinavian languages, however, the abundance of presentation constructions makes the distinction much clearer. In a presentation construction, the SUBJECT is semantically vacuous, and the highest-ranked Cf is typically the OBJECT[4]. As illustrated in (17), the C_b of the next utterance is most often realized by its SUBJECT, violating the grammatical parallelism constraint as formulated by Kameyama.

(17) a. Det sitter en mann på trappa.
[it$_{SUBJ}$] sits [a man$_{OBJ}$] on stairs-the
"A man is sitting on the stairs."

b. Han ser ut som om han venter på noen.
[he$_{SUBJ}$] looks out as if he waits on someone
"He seems to be waiting for someone."

The notion that it is really the TOPIC, and not necessarily the SUBJECT, which is the highest-ranked Cf makes sense with respect to the information structure of discourse. Referring (i.e., non-pleonastic) pronominal expressions are semantically underspecified and need to be linked to some kind of description (usually consisting of an NP) earlier in the discourse (disregarding indexicals for the moment). For the hearer, the most natural place to look for elaborative information that can establish the interpretation of the pronominal expression will be the entity that the discourse has most recently been "about", i.e., the TOPIC of the previous utterance. As shown in the next section, Michael Strube and Udo Hahn have presented alternative approaches that are based on or related to Centering Theory, but that use information structure rather than grammatical function as the guiding principle for Cf ranking.

Strube & Hahn: Functional centering

Strube and Hahn (1996, 1999) argue that, at least in languages with a relatively free word order, such as German, the Cf list should be ranked according to the familiarity

[4]Traditional grammars typically analysed such constructions using notions such as "formal" SUBJECT (corresponding to the SUBJECT in my analysis) and "logical" SUBJECT (corresponding to the OBJECT in my analysis). As pointed out by Sveen (1996), the view of the postverbal NP as a kind of SUBJECT is also supported by the assumption in linking theories (e.g., Grimshaw, 1990; Jackendoff, 1990) that agents/external arguments are always mapped onto subject position.

Sveen (1996) argues, however, that what was traditionally called the "real" SUBJECT functions as OBJECT syntactically, and he refers to various work in modern Norwegian grammar that supports this view. The tendency among traditional grammarians to label it as SUBJECT stems from confounding semantic roles and syntactic functions and a consequent desire to label a constituent with a semantic role of Agent as SUBJECT. Following Sveen, I analyse the postverbal NP in these constructions as an OBJECT.

of the discourse entity, rather than its syntactic function. They rank the entities using the familiarity scale defined by Ellen Prince (Prince, 1981, 1992).

Strube (1998) presents an approach, called the *S-list* approach, which functions in a way similar to centering, but which disposes with the notion of centers and centering transitions. Strube argues that his preference for hearer-old discourse entities performs a similar function to the preference for coreference with the C_b of the previous utterance (i.e., CONTINUATION or RETAIN transitions) that is found in the BFP algorithm. He demonstrates that, at least for his small test set of 576 anaphors, his functional approach clearly outperforms BFP (with 492 vs. 438 correct antecedents). This seems to indicate that Centering Theory might put too much weight on the single backward-looking center, at the expense of other hearer-old discourse entities.

LRC

Left-Right Centering (Tetreault, 1999, 2001) is an approach which is based on Centering Theory, but which aims to remedy the weaknesses of the Brennan et al. (1987) model that were pointed out by Kehler (1997).

Tetreault (2001) carried out an evaluation of various versions of his system compared to Hobbs (1978), the BFP algorithm (Brennan et al., 1987), and Strube (1998)'s S-list approach. The evaluation was performed on newspaper texts and fiction taken from the Penn Treebank (Marcus et al., 1993), and showed that LRC performed significantly better (with $p \leq 0.05$ using McNemar's test) than all the other systems on both text types. Hobbs' algorithm was the closest one to the LRC system in performance, with 76.8% vs. 80.4% accuracy on newspaper text and 80.1% vs. 81.1% on fiction, and these two systems performed much better than the two other systems (newspaper text: 59.4% for BFP and 71.7% for S-list; fiction: 46.4% for BFP and 66.1% for S-list).

Tetreault's results show that, for anaphora resolution, fiction is generally more difficult than non-fiction.

Although the results of Tetreault's evaluation look impressive for the LRC algorithm, it should be borne in mind that the evaluation was carried out on syntactically analysed, even manually annotated, material.

One of the strong points of Strube's S-list approach is that it does not rely on syntactic analysis at all; the propensity of an NP to function as the antecedent of an anaphor relies solely on its information status (hearer-new vs. hearer-old) and its position in the discourse. On the other hand, of course, it requires the ability to distinguish between hearer-new and hearer-old entities. Hence, since the systems are different with respect to whether the kind of information they need is provided manually, Tetreault's experiment does not necessarily tell us how the systems would compare in a real-world situation.

Byron's PHORA

Byron (2002) describes the PHORA system. Although not a Centering system in the traditional sense, it is based on a notion of salient discourse entities similar to the Centering systems described above. Most importantly, the system is equipped with a process of semantic filtering, which employs knowledge about the semantic restrictions of verbs. Unlike most other anaphora resolution work, Byron's system is evaluated on spoken dialogues, and it resolves anaphors with non-NP as well as NP antecedents. When evaluated on domains for which domain-specific resources are available, Byron's system achieves a 72% accuracy. If only domain-independent information is used, however, the performance drops to 51%, demonstrating the difficulties associated with resolving anaphors in spoken language, or anaphors with non-NP antecedents, or most likely a combination of these factors.

7.2.5 Factor/indicator-based approaches

Lappin and Leass

The procedure put forward by Lappin and Leass (1994) is called *RAP* (Resolution of Anaphora Procedure). It involves a number of components, the most well known being a set of *salience factors* to determine the most suitable antecedent of a given anaphor. The total collection of components is described as follows (see Lappin and Leass (1994) for further details):

- A syntactic filter on pronoun-NP coreference. This filter consists of six conditions in which intrasentential coreference between a non-reflexive and non-reciprocal pronoun and an NP can be ruled out on purely syntactic grounds.
- Tests for pleonastic pronouns, which by definition do not have antecedents. The tests consist of a set of syntactic patterns, along with sets of adjectives and verbs that, when they occur in these patterns, indicate the presence of pleonastic uses of *it*.
- Conditions for binding of reflexive and reciprocal pronouns. Lappin and Leass do not employ the principles of the mainstream binding theory developed by Chomsky (1981, 1986). Instead, they specify a filter consisting of six syntactic configurations in which a reflexive or reciprocal pronoun will be bound by an NP. This filter was originally presented by Lappin and McCord (1990), who argue that it covers roughly the same cases as conditions B ("a pronoun must never be bound within its domain") and C ("a referential expression must never be bound") of Chomsky's theory, but that it also applies straightforwardly to a wide variety of constructions which cannot be handled by the mainstream theory without special devices.

- A number of salience factors which are assigned to potential antecedents:
  - *Sentence recency*, assigned to all discourse referents that occur in the same sentence as the anaphor.
  - SUBJECT *emphasis*, assigned to all SUBJECTs.
  - *Existential emphasis*, assigned to predicate nominals in existential constructions ("There is *a fly* in my soup").
  - *Accusative emphasis*, assigned to DIRECT OBJECTs.
  - INDIRECT OBJECT *and oblique complement emphasis.*
  - *Head noun emphasis*, assigned to any NP which is not contained in another NP.
  - *Non-adverbial emphasis*, assigned to any NP not contained in an adverbial prepositional phrase demarcated by a separator. Note that, because of the constraint that the PP be demarcated by a separator, NPs in sentence-final PPs will typically be assigned this salience factor ("There are many famous museums in *this city*"), while those in sentence-initial position will not ("In *this city*, there are many famous museums"). Lappin and Leass do not discuss their motivations for making this distinction.

  Lappin and Leass state that each time a new sentence is processed, the weights of all salience factors that were assigned prior to the introduction of the new sentence are degraded by a factor of two, and that when the weight of a factor reaches zero, the factor is removed. Taking "degraded by a factor of two" to mean "divided by two", the weight cannot actually ever reach zero, so it is not entirely clear exactly when the factor is removed, but supposedly it happens when the weight reaches some sufficiently small level.

- A procedure for identifying equivalence classes of anaphorically linked NPs and computing the global salience value of the class as the sum of the weights of all salience factors assigned to at least one of the members of the class.

- A fairly complex procedure for selecting the most suitable antecedent, if any, for a given pronoun. The procedure makes use of all the other components. It also increases the weight of antecedent candidates that have the same syntactic function as the anaphor (i.e., it rewards syntactic parallelism), and decreases the weight of NPs following the anaphor (i.e., it penalizes cataphora).

The weights given to each of the salience factors are shown in Table 7.1. These weights were determined experimentally to be the ones that optimized the performance of the system.

Factor	Weight
Sentence recency	100
SUBJECT emphasis	80
Existential emphasis	70
Accusative emphasis	50
INDIRECT OBJECT and oblique complement emphasis	40
Head noun emphasis	80
Non-adverbial emphasis	50

Table 7.1: Weights given to the salience factors in Lappin and Leass's system.

The method reaches an accuracy of 86% on a corpus of 345 sentences containing 360 pronouns. The test sentences were randomly extracted from a corpus of computer manuals, subject to the condition that each selected sentence contained at least one non-pleonastic pronoun, and that the preceding sentence did not contain a pronoun.

A practical problem with Lappin and Leass's approach is that it relies on correct deep syntactic analysis of the text. Since deep syntactical analysis is hard (regardless of the particular formalism used), this is a difficult requirement to fulfil for any system that aims at robust, fully automatic anaphora resolution of running text. Furthermore, for most languages, tools for deep syntactic analysis do not even exist.

To alleviate these problems, Kennedy and Boguraev (1996) present an approach that reduces the syntactic analysis requirement from a deep hierarchical analysis to the kind of shallow dependency structure produced by a Constraint Grammar[5].

MOA/MARS

Ruslan Mitkov has made important contributions to the AR field through the knowledge-poor system presented in Mitkov (1998) and the further development of the system presented in Mitkov (2002). Mitkov calls the latter system *MARS* (an acronym for Mitkov's Anaphora Resolution System); the earlier system will be referred to here as *MOA* (Mitkov's Original Approach).

Both MOA and MARS are based on a set of so-called *indicators*, which work in a similar way to Lappin and Leass' factors (and to the input features used in machine learning approaches; cf. section 7.2.6 below). These indicators can be seen as functions that return a value based on certain aspects of the context in which the anaphor and/or a potential antecedent occur, e.g., whether the potential antecedent is the first NP in a sentence or whether it has been repeated several times in the current paragraph or in the document as a whole. The return values range from -1 to 2.

Although all of the systems presented here are knowledge poor compared to more knowledge-intensive systems such as Carter (1987) and Hobbs et al. (1993, 1988),

[5]Incidentally, this is the same grammar formalism that was used to analyse the Oslo Corpus; cf. chapter 3.

Mitkov puts particular emphasis on the small amount of linguistic knowledge required for his systems. Although the approach taken by Mitkov is similar to that of Lappin and Leass, Mitkov's systems do not rely on any form of binding theory for resolving reflexives, and MOA does not even make use of syntactic analysis, although the later MARS system does (albeit only in the form of a Constraint Grammar-based shallow dependency analysis, along with lemmatization and NP chunking, performed by Conexor's FDG parser (Tapanainen and Järvinen, 1997)).

On the other hand, while Mitkov's systems can certainly be considered knowledge poor, they nevertheless rely on highly language-specific and even genre-specific syntactic patterns (which even constitute one of the most important indicators for the system), as well as lists of so-called *indicating verbs* (which tend to be followed by NPs that are particularly salient and therefore prone to occur as antecedents). The Bulgarian version of the system even uses a small hand-crafted term bank containing domain-specific terms and phrases in which these terms occur (Mitkov, 2002, page 172). Hence, in some ways, Mitkov's approach actually seems fairly knowledge intensive compared to, for example, approaches based on Centering Theory, which do not assume anything about syntactic patterns or particular lexemes.

7.2.6 Statistical approaches

Unlike the symbolic methods that we have looked at so far in this chapter, statistical or machine learning approaches rely on training material. Statistical methods can be divided into supervised and unsupervised methods, depending on the kind of training material they use. Supervised methods require training material that has been annotated with information about the categories that the system is to handle, while this is not needed for unsupervised methods.

The most obvious advantage of unsupervised methods is that they do not require manually annotated training data. Furthermore, supervised systems tend to become tuned to the specific domain for which they were trained, and migration to different domains usually requires manual annotation of additional training data. Unsupervised systems, on the other hand, can more easily be adapted to different domains, since they do not rely on manual annotation.

Dagan and Itai (1990)

Dagan and Itai (1990) collect co-occurrence data from the Hansa corpus and use them in their unsupervised approach to find antecedents of the pronoun *it*. The authors find occurrences of the patterns SUBJECT-verb, verb-OBJECT, and adjective-noun, and count word co-occurrences in these patterns. Later, when the pronoun *it* is found in one of these patterns, it is replaced by each potential antecedent in turn, and

each antecedent candidate that has a co-occurrence count above a certain threshold is considered a likely candidate.

The test cases considered by Dagan and Itai are heavily restricted. The system is tested only on occurrences of *it* which have an NP antecedent. In other words, pleonastic uses and occurrences that have clauses or VPs as antecedents are manually filtered out from the test corpus. The set of test cases is further restricted to examples of *it* that occur in one of the above-mentioned syntactic patterns, which means that the experiments only test those cases where the collected statistical information is directly relevant.

Moreover, only anaphors with intrasentential antecedents were included, a fact which greatly reduces the number of antecedent candidates that have to be considered. On the other hand, results were reported only for those cases that were in fact ambiguous, and because only examples of *it* occurring after the 15th word of the sentence were included, there was nevertheless an average of 2.8 antecedent candidates per anaphor.

The system was tested on 59 cases of *it* from the Hansa corpus. Of these, only 38 contained constructions for which a sufficient number of cases was found in the training data (the threshold being set to five or more occurrences). Hence, the algorithm is said to have a coverage of 64% (38/59 cases). In 33 of the 38 cases that are covered by the algorithm, the correct antecedent candidate was included among those picked by the algorithm (i.e., those that had a co-occurrence count above the threshold). Based on this, the authors state that the algorithm has an accuracy of 87%. Furthermore, in 18 of these 38 cases, only one candidate was selected, yielding complete disambiguation.

Although the reported accuracy of Dagan and Itai's system is fairly high, it should be noted that their test set is very small by today's standards.

Aone and Bennett (1995) and McCarthy and Lehnert (1995)

Aone and Bennett (1995) and McCarthy and Lehnert (1995) both used decision trees, a supervised approach implemented in the C4.5 decision tree system (Quinlan, 1993), to select an antecedent among a set of candidates. Aone and Bennett applied their system to Japanese, while McCarthy and Lehnert (1995) focused on English. In both systems, anaphors and antecedents were represented by sets of features, and each anaphor was paired with each of its potential antecedents in turn to create a set of feature vectors that was used for training and testing the system.

The number of features employed by the systems differed considerably: Aone and Bennett's system contained 66 features, while that of McCarthy and Lehnert was restricted to only eight features. Nevertheless, there were clear similarities in the types of features that were used by the systems. Both systems used lexical, semantic, and positional features (Aone and Bennett also included syntactic features). Furthermore,

both systems employed both unary features (i.e., features that only involve either the anaphor or the antecedent) and binary features (i.e., features that involve both the anaphor and the antecedent).

The decision tree classifiers were trained to classify pairs of NPs as *anaphor–antecedent* pairs or non-*anaphor–antecedent* pairs[6]. In order to learn such a binary classification task, a supervised learner needs to be presented with both positive and negative examples.

Aone and Bennett ran two conditions. In the first condition (the "anaphoric chain" condition), an anaphor was paired with all NPs that occurred earlier in its anaphoric chain to create positive training examples (i.e., if B had A as its closest antecedent and C had B as its closest antecedent, C was paired with both B and A in turn to make two positive examples). In the second condition, an anaphor was only paired with its closest antecedent. McCarthy and Lehnert (1995) always paired an anaphor with all of the NPs in its anaphoric chain. In both papers, the negative training examples for an anaphor were created by pairing it with all possible antecedents except for those that occurred in its anaphoric chain.

The best performance reported by Aone and Bennett is an $F_{\beta=1}$ of 77.43, while McCarthy and Lehnert's system reaches a maximum $F_{\beta=1}$ of 86.5. Despite the similarities between the systems, however, the results are not comparable, since the systems are applied to different languages (for example, the anaphors handled by Aone and Bennett's system for Japanese include zero pronouns, which do not (normally) exist in English).

It should also be noted that the $F_{\beta=1}$ of McCarthy and Lehnert's system on the MUC-6 test corpus was only 47.2. Soon, Ng, and Lim (2001), in discussing this result in comparison to their own (cf. section 7.2.6), point out that McCarthy (1996) explains the low MUC-6 score by the fact that the system only tries to resolve references to person and organization entities, and furthermore that it does not extract nested noun phrases (i.e., NPs that are embedded in other NPs). This restricted focus is believed to lead to a relatively low recall on the MUC-6 data set.

Ge, Hale, and Charniak (1998)

Ge et al. (1998) take a probabilistic, unsupervised approach to the task of pronominal anaphora resolution. They identify a number of potentially useful features that can be extracted from the training and test data: the distance between the pronoun and its antecedent, the syntactic environment of the pronoun, the words that constitute a (potential) antecedent phrase, and the number of times a (potential) antecedent has been mentioned so far in the text. Using a small portion of the Penn Treebank (Marcus

[6] In one condition, Aone and Bennett's system was also trained to select among several *types* of antecedents, but I am focusing on the binary classification task here.

et al., 1993) as training data, the authors calculate the probability of a given NP being the antecedent of a certain pronoun given the values for each of these features that can be extracted from the context.

With respect to the last factor (the mention count), the authors point out that it helps them identify the topic of a particular segment of the discourse, and that this is reminiscent of a Centering Theory approach to AR (e.g., Brennan et al., 1987), in which a continued topic is the preferred choice of antecedent.

In their experiments, Ge et al. focus on the pronouns *he*, *she*, and *it*, and they exclude pleonastic (i.e., non-referential) cases of *it* from the test cases. Running 10-fold cross-validation on about 94,000 words from the Penn Treebank, they obtain an accuracy of 82.9%. Applying an unsupervised method of acquiring gender information from unannotated text, they are able to boost the performance of their AR system to 84.2%.

Cardie and Wagstaff (1999)

Cardie and Wagstaff (1999) introduced an unsupervised approach that treated AR as a clustering task rather than a classification task. In their system, each noun phrase is represented as a feature vector. The aim of the system is to cluster these vectors into equivalence classes, with each class representing a single discourse entity. Some features are context independent and only represent aspects of the NP itself, while others are context dependent, which in this connection is taken to mean that they take into account the relationship between the NP and other NPs in the context.

While unsupervised learning tends to require far less manual work than supervised learning, the usual trade-off is that it results in a lower performance score, and anaphora resolution is no exception in this respect. Cardie and Wagstaff's system obtained an $F_{\beta=1}$ of 53.6 on the MUC-6 coreference data. Although this means that the system outperforms many of the manually crafted systems that participated in MUC-6, as well as the only one based on machine learning, it is nevertheless beaten by later supervised systems that have been tested on the same data. These systems are described in the following sections.

Soon et al. (2001)

Soon et al. (2001) present another AR system that uses a decision tree to classify NP as antecedents or non-antecedents of a given anaphor. They do tokenization, morphological processing, part-of-speech tagging, named entity recognition, nested noun phrase extraction (i.e., extraction of NPs that are embedded within other NPs, such as *his* in *his long-range strategy* and *Eastern* in *Eastern's parent*), and semantic class determination, before feeding the resulting NPs, or *markables*, into the decision tree system for classification. The preprocessing steps serve to determine the boundaries

of the markables in the text, and to provide information that can be utilized when composing the features that are used by the classifier.

Like Aone and Bennett (1995) and McCarthy and Lehnert (1995), Soon et al. use both unary features (which only regard either the anaphor or the antecedent) and binary features (which take both the anaphor and the antecedent into account). A total of 12 features are included. Since Soon et al.'s work was the primary inspiration for the machine learning approach to Norwegian AR described in chapter 8, I will present the features used by these authors in some detail. The following list is a somewhat abbreviated quotation of the feature descriptions given in the paper (Soon et al., 2001, pp. 524–526), with my omissions and substitutions of parts the original text indicated by brackets; i denotes the anaphor, while j represents an antecedent candidate.

- *Distance Feature (DIST)*: Its possible values are 0,1,2,3,.... This feature captures the distance between i and j. If i and j are in the same sentence, the value is 0; if they are one sentence apart, the value is 1; and so on.

- *i-Pronoun Feature (I_PRONOUN)*: Its possible values are true or false. If i is a pronoun, return true; else return false. Pronouns include reflexive pronouns (*himself, herself*), personal pronouns (*he, him, you*), and possessive pronouns (*hers, her*).

- *j-Pronoun Feature (J_PRONOUN)*: Its possible values are true or false. If j is a pronoun (as described above), then return true; else return false.

- *String Match Feature (STR_MATCH)*: Its possible values are true or false. If the string of i matches the string of j, return true; else return false. We first remove articles (*a, an, the*) and demonstrative pronouns (*this, these, that, those*) from the strings before performing the string comparison. Therefore, *the license* matches *this license*, *that computer* matches *computer*.

- *Definite Noun Phrase Feature (DEF_NP)*: Its possible values are true or false. [...] If j is a definite noun phrase, return true; else return false.

- *Demonstrative Noun Phrase Feature (DEM_NP)*: Its possible values are true or false. A demonstrative noun phrase is one that starts with the word *this*, *that*, or *those*. If j is a demonstrative noun phrase, then return true, else return false.

- *Number Agreement Feature (NUMBER)*. Its possible values are true or false. If i and j agree in number (i.e., they are both singular or both plural), the value is true; otherwise false. [...]

- *Semantic Class Agreement Feature (SEMCLASS)*: Its possible values are true, false, or unknown. [...] the semantic class for every markable extracted is the

> [most frequent WordNet] sense of the head noun of the markable. [...] If the selected semantic class of a markable is a subclass of [one of a number of predefined semantic classes], then the semantic class of the markable is [that predefined class]; else its semantic class is "unknown".
>
> The semantic classes of markables i and j are in agreement [and the feature returns true] if one is the parent of the other (e.g., *chairman* with semantic class "person" and *Mr. Lim* with semantic class "male"), or they are the same (e.g., *Mr. Lim* and *he*, both of semantic class "male"). [...] If either semantic class is "unknown", then the head noun strings of both markables are compared. If they are the same, return true; else return unknown.

- *Gender Agreement Feature (GENDER)*: Its possible values are true, false, or unknown. [Gender is determined using designators and pronouns such as *Mr.*, *Mrs.*, *she*, and *he*, or by semantic class.] If the gender of either markable i or j is unknown, then the gender agreement feature value is unknown; else if i and j agree in gender, then the feature value is true; otherwise its value is false.

- *Both-Proper-Names Feature (PROPER_NAME)*: Its possible values are true or false. A proper name is determined based on capitalization. [...] If i and j are both proper names, return true; else return false. [...]

- *Alias Feature (ALIAS)*: Its possible values are true or false. If i is an alias of j or vice versa, return true; else return false. [Aliases are found using string match (e.g., *Mr. Simpson* and *Bent Simpson*) or acronym match (e.g., *IBM* and *International Business Machines Corp.*).] [...]

- *Appositive Feature (APPOSITIVE)*: Its possible values are true or false. If j is in apposition to i, return true; else return false. For example, the markable *the chairman of Microsoft Corp.* is in apposition to *Bill Gates* in the sentence *Bill Gates, the chairman of Microsoft Corp.,...* [...]

Positive training instances were created by pairing an anaphor only with its closest antecedent, leading to a much smaller set of positive examples than in McCarthy and Lehnert (1995)'s system and in the "anaphoric chain" condition of Aone and Bennett (1995). However, although such a reduction in the number of positive examples helps to reduce the computational resources needed for the system, the real problem with the data creation procedure in earlier work had been that they created far too many *negative* examples, yielding a training set that was heavily skewed towards the negative class (see Hoste (2005, chapter 7) for further discussion on this topic).

Soon et al. try to alleviate this problem by composing negative examples using only those markables that lie between an anaphor and its closest antecedent (instead of all earlier markables that are not part of the same anaphoric chain). Although this is

bound to reduce the skewedness of the training set to a considerable extent, Soon et al. nevertheless report little or no improvement compared to experiments where they use McCarthy and Lehnert (1995)'s methods for creating training instances, indicating that such a skewedness does not in fact pose a problem for a decision tree classifier.

When tested on the MUC-6 data, Soon et al. (2001)'s system obtains an $F_{\beta=1}$ of 62.6, which is far better than both McCarthy and Lehnert's earlier supervised system and the unsupervised system of Cardie and Wagstaff (1999). On the MUC-7 test set, it obtains an $F_{\beta=1}$ of 60.4.

Interestingly, when each feature is tested separately, only the ALIAS, STR_MATCH, and APPOSITIVE features give non-zero F-measures, and, if only these three features are used together, they yield an F-measure which is close to that of the whole feature set (60.3 vs. 62.6 on the MUC-6 test data and 59.4 vs. 60.4 on the MUC-7 test data). Furthermore, among these three features, ALIAS and STR_MATCH yield much better results than APPOSITIVE, and the combination of these two features alone also performs well compared to that of the entire feature set (58.0 vs. 62.6 on the MUC-6 data and 58.1 vs. 60.4 on the MUC-7 data). These results show that a few simple features, whose values can easily be extracted from tokenized text, can yield a system which holds up surprisingly well in comparison to systems that require much more sophisticated linguistic information.

Ng and Cardie (2002b) and Ng (2005)

Ng and Cardie (2002b) extend Soon et al.'s set of 12 features with an additional 41 features. Furthermore, they compare two different techniques for selecting a particular antecedent among those that have been classified by a machine learner as suitable antecedent candidates: the *closest-first* technique, which selects the closest antecedent, and the *best-first* technique, which selects the candidate with the highest confidence value. Ng (2005) also tests these two techniques and includes a third one, *aggressive merge*, in which the anaphor is merged with all its preceding coreferent NPs.

Somewhat surprisingly, it turns out that the additional features introduced by Ng and Cardie lead to *decreased* performance, in spite of the fact that the authors use learning frameworks (C4.5 and RIPPER) that should be able to pick out and emphasize those features that are useful for solving the task. On the other hand, they find that their best-first clustering approach leads to some improvement over the closest-first approach.

However, the available evidence, both with respect to the feature set and with respect to the choice between a closest-first and a best-first clustering technique, is inconclusive. Ng (2005) performs a comparison of systems using different combinations of instance creation methods, feature sets, machine learners, and clustering algorithms. Although Ng and Cardie (2002b) report a decreased performance with their extended

feature set, this set performs best in almost all cases in Ng (2005)'s study. On the other hand, in spite of the fact that Ng and Cardie (2002b) show improved performance for their best-first system as compared to the closest-first approach taken by Soon et al. (2001), none of the experiments conducted by Ng (2005) show this approach to be the best one (of the six experiments reported, closest-first wins two and aggressive-merge wins the remaining four).

These results might seem to suggest that the optimal choice of feature set and clustering technique depends entirely on text type. However, some indication that this is not so is given by Ng (2005)'s results, in which Ng and Cardie's feature set performs best in all but one case. Furthermore, the instance creation method and clustering algorithm employed by McCarthy and Lehnert provides the best results in the majority of cases. Not coincidentally, perhaps, these are also the choices that require the most resource-intensive computations, since they involve the creation of training instances for *all* anaphors paired with *all* NPs in the text, and clustering of an anaphor with *all* of its antecedents. Thus, this can be seen as yet another instance of the common trade-off between efficiency and performance.

Other work

Finally, there are a number of other works on machine learning approaches to AR which should be mentioned.

Ng and Cardie (2002a) use a decision tree to classify all NPs in a text as either anaphoric or non-anaphoric, and incorporate this classifier into the coreference resolution system presented by Ng and Cardie (2002b).

Applying the anaphoricity classifier indiscriminately to all potential anaphors is actually detrimental to the performance of the coreference resolution system. The authors point out, however, that string matching and aliasing features have proven to be very strong indicators of coreference in their coreference resolution system. They then propose to bypass the anaphoricity classifier in those cases where these features match. Applying this additional constraint, the combined system (coreference classifier + coreference resolution system) significantly outperforms the original system.

Ng and Cardie (2003b) compare co-training, self-training with bagging, and weakly supervised Expectation-Maximation (EM) on the coreference task. They conclude that self-training performs better than co-training on this task, because there is no natural way to split the available features into the set of multiple separate but redundant views of the data that is required for co-training. EM by itself performs worse that co-training, but in combination with feature selection it also outperforms co-training.

In Ng and Cardie (2003a), the same authors investigate an alternative solution to the lack of a natural way to split the features into multiple views. Drawing on work by Goldman and Zhou (2000) and Steedman, Osborne, Sarkar, Clark, Hwa, Hockenmaier,

Ruhlen, Baker, and Crim (2003), they apply two different learning algorithms—Naive Bayes and decision list learners—to the same data. The underlying hypothesis is that the differences between the learning algorithms, which result from their different data representations and their different biases, can substitute for having different views of the data. This approach also outperforms co-training on the coreference task.

Strube, Rapp, and Müller (2002) use a decision tree-based approach which includes a feature that expresses the minimum edit distance between anaphor and antecedent. The minimum edit distance measures the minimum number of substitutions, insertions, and deletions that are needed in order to transform one string into another. The inclusion of this feature leads to significant performance improvements, particularly on definite NPs and proper names.

The system described by Strube and Müller (2003) also employs a decision tree, but, like the work of Byron (2002) (cf. section 7.2.4), it differs from most other work on anaphora and coreference resolution in two respects. Firstly, the system is applied to spoken dialogues, rather than to written material. Secondly, because a significant amount of anaphors in spoken language have non-NP antecedents or no antecedents at all (Botley, 1996; Byron, 2002; Byron and Allen, 1998; Eckert and Strube, 2000), Strube and Müller (2003) try to handle non-NP antecedents as well as NP antecedents. As is to be expected, the performance level of their system is significantly lower than state-of-the-art results on NP antecedents in written texts, reaching an F-score of 47.42% on a dialogue corpus of 30,810 tokens. The authors point out that, although the performance is fairly low, it nevertheless approaches that of Byron (2002)'s rule-based system when the latter is evaluated without domain-specific knowledge.

Hoste (2005) applies both memory-based learning (TiMBL) and rule induction (RIPPER) to English and Dutch anaphora resolution. She uses genetic algorithms to perform exhaustive optimization of both the feature set and the parameter settings that are used for each of the two machine learning mechanisms, and shows that the optimizations make the F-scores of the two mechanisms converge. She concludes that parameter optimization is a very important step in machine learning experiments, and that many of the differences between machine learning methods that have been reported in the literature result from inadequately optimized systems and therefore should not be taken at face value.

Hoste and van den Bosch (2007) apply a post-processing step to the systems presented in Hoste (2005). The post-processing is based on Levenshtein distance (Levenshtein, 1965), which is a special case of minimum edit distance in which each edit operation is weighted equally with respect to its contribution to the total distance (cf. Jurafsky and Martin, 2000, pp. 154–155). Coreference chains are expressed as "words" written in various small alphabets. The alphabets consist of symbols which represent types of NPs, the distance between them, and the way they are linked. The

post-processing step is then formulated as an error correction task in which the coreference chains produced by the coreference resolution system are corrected based on their Levenshtein distance from the chains that occur in the training corpus.

Hoste and van den Bosch's best results are obtained using an alphabet that represents each individual NP as a separate symbol, whether or not the NP is part of the coreference chain. For example, if a chain contains two coreferential common nouns (represented by L), and there are three other NPs (represented by O) between these nouns that do not belong to the same chain, the pattern will be represented as LOOOL. Although the performance improvement obtained is not large—the precision increases from 67.5 to 69.9 and the F-score from 51.3 to 51.8—it is interesting, and somewhat surprising, to see that some improvement can be achieved using such a simple representation of coreferential chains. Apparently, the patterns capture some inherent structure in coreferential chains, despite the fact that they ignore the syntactic complexities of natural language.

Hendrickx, Hoste, and Daelemans (2008) include two types of semantic information in their coreference system for Dutch: one the one hand, synonym and hypernym relations taken from EuroWordNet (Vossen, 1998), and, on the other, semantic clusters of nouns derived by van de Cruys (2005). When classification is performed by TiMBL, using the clustering information or both types of information in combination improves performance, while the WordNet relations in themselves do not. When MaxEnt is used for classification, applying either information type by itself actually decreases performance, but a combination of WordNet relations and clustering performance leads to improved results. Hendrickx et al. also show a small positive effect of complementing shallow syntactic function information (such as whether an NP is a SUBJECT or an OBJECT) with a fuller dependency analysis.

Manfred Klenner and Étienne Ailloud (Klenner, 2007; Klenner and Ailloud, 2008) have worked on a particular problem of the clustering techniques used in machine learning-based AR systems. In general, such systems do not prevent inconsistent clustering of coreferent and non-coreferent pairs of markables. In other words, if the system makes the following decisions:

- markables A and B are *not* coreferent
- markables C and D are *not* coreferent
- markables A and C *are* coreferent

then there is generally nothing to prevent the system from classifying markables A and D as coreferent, even though such a decision would be inconsistent with the decisions already made. Klenner and Ailloud use a technique called Integer Linear Programming (ILP) to filter out such inconsistencies from the coreference chains.

Over the last few years, Xiaofeng Yang and co-workers have made a series of contributions to the field of statistical AR. Yang, Zhou, Su, and Tan (2003) introduce a so-called *twin-candidate model* in which antecedent candidates compete against each other in pairs and the one that wins the most competitions is selected as antecedent. Since the number of competitions would be prohibitively large if all antecedent candidates were allowed to compete, only a selected set of candidates are considered. For pronominal anaphors, this set consists of all NPs in the current sentence and the two previous sentences that agree with the anaphor in number, gender, and person (or in earlier sentences if no such candidates can be found). In the case of non-pronouns, Yang et al.'s description of the initial selection process for the training set is unclear[7], but at least it is clear that only non-pronominal candidates are considered, and that during resolution all candidates are classified using a single-candidate classifier and all candidates with confidence value below 0.5 are removed.

Yang et al. report that their system outperforms those of Strube (1998), Ng and Cardie (2002b), and Connolly, Burger, and Day (1997) on the MUC-6 and MUC-7 test corpora, with the biggest improvement showing on pronoun (as opposed to non-pronoun) resolution.

Yang, Su, Zhou, and Tan (2004a) take as their starting point the system described by Soon et al. (2001) and add to it a number of features that, for each antecedent candidate, describe the antecedent of the candidate (if any such antecedent exists). In other words, they incorporate a small part of the coreference chain into the feature vector. Yang et al. show that this improves the performance of the system on the MUC-6 and MUC-7 test corpora.

A similar approach is applied by Yang, Su, Zhou, and Tan (2004b) to the task of general coreference resolution. During training, the representation of an antecedent NP includes a representation of the cluster of coreferential NPs to which the antecedent belongs. During resolution, the antecedent candidates are linked to each previously established coreference cluster in turn, and the probability that the candidate belongs to a cluster is assessed by a decision tree. When tested on data from the biomedical domain, this approach is shown to outperform the traditional approach which only considers individual antecedent candidates.

Yang, Su, and Tan (2005) extract possessive–noun, SUBJECT–verb, and verb–OBJECT relationships from the Web and use this information to improve pronoun resolution, both for a traditional single-candidate model and for the twin-candidate model described by Yang et al. (2003), mentioned earlier in this section. The au-

[7] Their description goes as follows (Yang et al., 2003, p. 130):

(a) Add all the non-pronominal antecedents to the initial candidate set.

(b) For each candidate added in [(a)], add the non-pronouns in the current, the previous and the next sentences into the candidate set.

thors show that acquiring such information from the Web yields better results than extracting them from a 76-million-word corpus of *Wall Street Journal* articles. They also demonstrate that not only does the twin-candidate model perform better than the single-candidate model (as had already been shown by Yang et al. (2003)), but the twin-candidate model is also able to make better use of the semantic information. However, the authors conclude that the performance improvements are limited to the neuter pronoun *it*; for other pronouns, no improvement is achieved.

Unlike their earlier work, which is based on decision trees, the work described by Yang, Su, and Tan (2006) employs a support vector machine and uses syntactic parse trees as part of the input to the pronoun resolution task. The trees are used directly as structured features, rather than extracting different kinds of syntactic relations into separate features, as would be the traditional approach, and this means that all of the information encoded in the tree is available to the SVM. The authors show that inclusion of the syntactic feature leads to significant improvement on the pronoun resolution task.

Finally, Yang and Su (2007) describe a technique for automatically acquiring text patterns that may contribute useful information about semantic relations to the coreference resolution task. They extract a set of patterns from Wikipedia[8], provide them as input to a decision tree-based coreference system along with the features described by Ng and Cardie (2002b), and state that the pattern-based semantic information leads to a significant performance boost.

7.2.7 Related work on Norwegian AR

Until recently, very little work had been carried out on anaphora resolution for Norwegian, but a couple of Norwegian AR systems, in addition to the one presented in chapter 8, have surfaced in the last couple of years.

Holen (2006)

Holen (2006) presents a factor/indicator-based system, *ARN*, which is heavily influenced by Mitkov's work (Mitkov, 1998, 2002) as well as the work of Lappin and Leass (Lappin and Leass, 1994). Taking the factors or indicators presented by these authors as a starting point, and testing how well each of them performs on Norwegian data, Holen arrives at the following set of factors for her system:

- **Number/gender/animacy**: Match or mismatch between anaphor and antecedent candidate on number, gender, and animacy is awarded points according to Table 7.1.

[8] `http://http://en.wikipedia.org/wiki/Main_Page`

- **Reference proximity**: Proximity between anaphor and antecedent is awarded in the following way:
  - Antecedent found in the same sentence: 100 points
  - Antecedent found in the previous sentence: 50 points
  - Antecedent found in earlier sentences: 0 points
- **Boost pronoun**: Pronominal candidates are awarded 75 points.
- **Direct object preference**: Direct object candidates are awarded 50 points.
- **Adverbial phrase penalization**: Candidates in adverbial phrases are penalized with -50 points.
- **Syntactic parallelism**: Candidates with the same syntactic function as the anaphor are awarded 50 points.
- **Section heading preference**: Candidates that occur in the heading of the section in which the anaphor occurs are awarded 50 points.
- **Indefiniteness penalization**: Indefinite candidates are penalized with -25 points.

Importantly, unlike the systems for English, Holen has not included SUBJECT preference as a factor. She has shown that information structure is expressed differently in Norwegian and English, with the former frequently using presentation constructions, which have a non-referential pronoun as SUBJECT.

Pronouns			***den***	***han***	***hun***
Nouns	Class 1 *not human*	m	100	-100	-100
		f	100	-100	-100
		n	0	-100	-100
	Class 2 *human of indeterminable gender (male?)*		-50	100	75
	Class 3 *human male*		-50	100	0
	Class 4 *human female*		-50	0	100

Figure 7.1: Points awarded based on gender and animacy in ARN. Adapted from Holen (2006) (points for pronouns that are not evaluated (*De* and *de*) have been removed).

Like the systems presented in the current work, ARN is tested on data from the BREDT project. However, Holen uses an earlier version of the data, which has been

annotated according to guidelines that are different from those used in the current work. Furthermore, the fiction data used here only constitute half of Holen's data; the rest of her material consists of newspaper text[9].

Similarly to my own systems, Holen's system aims at handling third-person pronouns. However, she does not try to handle the neuter third-person singular pronoun *det* "it (neut.)" or reflexives or possessive pronouns (which are not part of the coreference chains in her data). Furthermore, the plural pronoun *de* "they" is excluded from her evaluation procedure[10]. Finally, unlike the present systems, Holen includes the second-person singular polite pronoun *De* "you". Her system obtains an accuracy of 70.5%, reaching 70.2% on the fiction material alone.

In order to compare Holen's approach to the machine learning approach developed in this thesis, I have reimplemented ARN as a module in my own anaphora resolution system. This reimplementation, which also includes a number of improvements to the original system, is presented in section 8.10.

Lech and de Smedt (2007)

Lech and de Smedt (2007) present some work on improving coreference resolution in Norwegian texts through the use of *ontologies*, defined as a specification of a conceptualization in a given domain. Due to their focus on ontological knowledge, Lech and de Smedt classify their work as knowledge based, but it is nevertheless data-driven since they aim at extracting the required knowledge automatically from corpora.

Lech and de Smedt use the Oslo-Bergen tagger (cf. section 3.3) to do morphological and syntactic tagging of a set of Norwegian newspaper texts, all of which concern very similar subjects. Subsequently, they proceed to extract verbs together with their corresponding SUBJECTs using the SPARTAN system (Velldal, 2003) and compute similarities between SUBJECTs based on their co-occurrence distribution with verbs. They use this information to enhance a manually constructed ontology of concepts in the particular domain that their texts belong to. Finally, they integrate this ontology into a system for Norwegian coreference resolution, and this is claimed to improve the results of the system, although no quantitative results are provided.

7.2.8 Other Scandinavian systems

With respect to other Scandinavian languages, Hassel (2000) presents an AR system for Swedish which is based on a knowledge-poor, rule-based algorithm developed by Fraurud (1992). Hassel's system is integrated into the SweSum automatic text summarizer (Dalianis, 2000) and is reported to improve the results of the summarizer.

[9] The newspaper material has not been annotated in accordance with the more recent guidelines, which is why it is not included in my data.

[10] Holen (2006) does not explicitly state that the plural pronoun is excluded from consideration, but this has been established in personal communication with the author.

Navarretta (2002) uses a combination of information structure and Centering Theory to resolve pronominal anaphora in Danish, while Nararretta (2004) describes the DAR system, which resolves both individual and abstract anaphora (anaphors that refer to abstract entities denoted by verbal phrases, clauses, or discourse segments) and outperforms competing algorithms on the same Danish data.

7.3 Conclusions

We have seen that all of the systems described in this chapter use linguistic information directly or indirectly. Many of the factors that have been explicitly mentioned have been successful across systems and across languages. However, both linguistic and extra-linguistic factors are important for the success of a system. We can summarize some important points:

- The language involved is an important factor (cf. section 7.2.7).
- Genre plays a role (cf. the description of Tetreault's experiment in section 7.2.4).
- Using irrelevant or too much information may be detrimental (cf. the work by Ng and Cardie described in section 7.2.6).
- Optimizing the algorithm may improve the results (cf. Hoste's work as presented in section 7.2.6).

Chapter 8

Pronominal Anaphora Resolution for Norwegian

8.1 Introduction

Chapter 7 presented some important work on knowledge-poor anaphora resolution (AR) methods that has been carried out over the past 30 years, with a particular focus on what has happened during the last decade. In this chapter I will present my own work on Norwegian AR, which relies mainly on the use of machine learning approaches.

I have presented some early versions of my system in Nøklestad, Johansson, and van den Bosch (2004), Nøklestad and Johansson (2005), and Nøklestad, Reigem, and Johansson (2006). However, the system presented in this chapter is substantially improved with respect to those versions: the current system uses a different set of machine learning features; the implementation of the filters (cf. section 8.5.1) has been tweaked; and a number of support modules have been developed (i.e., the named entity recognizer, the PP attachment disambiguator, and the animacy detection procedures presented in earlier chapters).

The main focus of this chapter is on my anaphora resolution system for Norwegian, implemented as a machine learning approach, primarily using TiMBL. Unlike earlier approaches, such as Hoste (2005), I have implemented three different classifiers for different kinds of pronouns, on the assumption that they have very different conditions for their distribution. This turns out to cause significant performance improvements. I have also introduced machine learning features that approximate principles A and B of Chomsky's Binding Theory (Chomsky, 1981, 1986)[1]. The first of these features re-

[1]Ng and Cardie (2002b) also include some representation of binding constraints in their system. However, they use only a single feature, and this feature indicates whether an anaphor–antecedent

ceives a very large gain ratio weight, indicating that it provides important information to the system, although removing this feature does not by itself lead to a significant performance decrease, possibly because its function is then taken over by other features. Finally, and importantly, I have included features that take advantage of the results from the three support modules that I have implemented for this purpose: a named entity recognizer, an animacy detector, and a PP attachment disambiguator.

Although the main focus of this chapter is on the machine learning approach, I also describe two alternative approaches. One is inspired by Centering Theory (cf. section 7.2.4), while the other one is a factor/indicator-based algorithm (cf. section 7.2.5).

My reasons for developing these alternative approaches are twofold. Firstly, they provide an opportunity for a direct comparison of approaches based on the different trends that have dominated knowledge-poor AR during recent decades. (I ignore here the kind of knowledge-poor systems presented by Hobbs (1978) and Tetreault (1999, 2001), because they rely on correct deep syntactic analyses, and such data are as yet not available for Norwegian.) Secondly, these implementations make it possible to experiment with mixed models that may potentially perform better than either "pure" machine learning-based methods or methods based on fixed rules or parameter settings. Such experiments are further facilitated by the fact that the various approaches are actually implemented as different modules in the same overall AR system.

8.2 Anaphors handled by the system

The AR algorithms presented in this thesis deal with pronominal anaphora, i.e., pronouns and reflexives. As is customary in AR work, only third-person pronouns are included, since first- and second-person pronouns are typically used deictically and often do not have any antecedent in the text.

The following pronouns are handled by the system:

- Personal pronouns
  - Singular
    - Animate pronouns:
      - Non-possessive: *han/ham* "he/him"; *hun/henne* "she/her"
      - Possessive: *hans* "his", *hennes*, "her(s)"
    - Inanimate pronouns:
      - Non-possessive masculine/feminine: *den* "it (masc./fem.)"
      - Possessive masculine/feminine: *dens* "its (masc./fem.)"

link would violate principles B and C of the Binding Theory, while the present work uses separate features that indicate whether the link adheres to principles A or B, respectively.

        - Non-possessive neuter: *det* "it (neut.)"
        - Possessive neuter: *dets* "its (neut.)"
    - Plural:
        - Non-possessive: *de/dem* "they/them"
        - Possessive: *deres* "their(s)"

- Reflexives
    - Non-possessive: *seg* "himself/herself/itself/themselves"
    - Possessive: *sin/sitt/sine* "his/her/its/their"

As the overview shows, I am using the traditional term "possessive pronoun" for *hans*, *hennes*, *dens*, *dets*, and *deres*, although these are actually analysed as determiners by the Oslo-Bergen tagger in accordance with Faarlund, Lie, and Vannebo (1997). For the anaphora resolution task, it does not matter whether these elements are called pronouns or determiners, and in order to emphasize the parallelism in the set of elements handled by the AR system, I will use the traditional terms.

8.3 Corpus

For developing and testing the different AR implementations, I use a part of the Oslo Corpus of Tagged Norwegian Texts (Hagen et al. (2000b); cf. chapter 3), which has been automatically annotated for part-of-speech tags and syntactic categories by the Oslo-Bergen tagger (Hagen et al., 2000a) and manually annotated for anaphoric and cataphoric relations by participants in the BREDT project. This subcorpus consists entirely of fiction material. The corpus format was shown in chapter 3, and is repeated here in Figure 8.1 on page 248.

The BREDT corpus has been annotated with a wide range of anaphoric relations (Borthen, 2004; Borthen, Johnsen, and Johansson, 2007): coreference, metonymy, intensional reference, reciprocal reference, bound variables, subset relations, superset relations, excluding reference, inalienable possession (body parts), family relations, and identity of sense.

Like most earlier AR systems, the system presented in this thesis only deals with the first of these relation types, i.e., coreference. This is motivated by the assumption that a machine learner will have a greater chance of success if it is trained on a single relation type at a time. The coreference relation is fairly straightforward to identify in most cases and therefore seems like a good starting point for implementing an AR system. Hence, in the training and test data presented to the system, only coreference relations are marked.

The BREDT corpus consists of 25,451 tokens, 2388 of which are pronouns that belong to one of the types handled by the system (i.e., the types listed in section 8.2). Of these 2388 anaphor candidates, 1768 (74.0%) are marked with a coreferential antecedent in the text. The corpus has been split in half. The first half is used for training the machine learning algorithms and for tuning the centering and factor-based implementations, while the second half serves as a development corpus which is used for selecting features for the machine learning algorithms and for a first round of tests of the various approaches (see section 8.7.6 for the results of testing on a completely separate test corpus). Table 8.1 gives an overview of the numbers of anaphors and antecedents in the corpus, while Table 8.2 shows the number of pronouns belonging to each of the pronoun types that are handled by the system.

Note that, in the machine learning approaches, reflexives are handled by the same classifiers as *han* and *hun*, and these pronouns are therefore lumped together in Table 8.2. Thus, any classifiers that are referred to as *han/hun* classifiers in this chapter also handle resolution of reflexives. However, in order to make my evaluations compatible with those of Holen (2006), reflexives are not included in any of the evaluations, since reflexives were not included in the coreference chains in Holen's data and were therefore not handled by her system.

The present system is not aimed at handling cataphora. Consequently, cataphors are excluded from the corpus and are not included in the evaluation of any of the approaches.

	Anaphors total	Anaphors with antecedents
Training corpus	1200	882 (73.5%)
Development corpus	1188	886 (74.6%)
Total	2388	1768 (74.0%)

Table 8.1: Total number of anaphors and number of anaphors with antecedents in the corpus.

han/hun/REFL	*den*	*det*	*de*	Total
1440	74	610	264	2388

Table 8.2: The number of pronouns of the relevant types in the training and development corpora.

8.4 Overall architecture

The different AR algorithms are implemented as alternative strategies that are plugged into the same overall architecture. The overall system implements the following steps for performing AR on a given text:

- The text is processed by the Oslo-Bergen tagger (cf. section 3.3), which tokenizes the text, tags it with part-of-speech and morphological information, and performs syntactic dependency parsing.

- All markables in the text are identified. Markables are defined to be common nouns, proper nouns, and pronouns. All markables can function as antecedents. The system also has the ability to treat all markables as anaphors; however, in this thesis only pronominal anaphors are considered.

- Anaphora resolution. For each anaphor, a single antecedent is identified, defined as the closest markable that is coreferent with the anaphor. The primary technique used to perform this step is machine learning (memory-based learning, maximum entropy modelling, or support vector machines), but alternative strategies built on Centering Theory or factor/indicator-based approaches are also tested.

8.5 The machine learning approach

This section describes the approach to which I have devoted the majority of my work, that is, an AR system for Norwegian that relies on machine learning. Being a machine learning approach, it is placed within the tradition that was established by the work of Aone and Bennett (1995), McCarthy and Lehnert (1995), and Soon et al. (2001), with later improvements and evaluations presented by work such as Ng and Cardie (2002b), Ng and Cardie (2003a), Ng (2005), and Hoste (2005).

As described in chapter 7, most earlier work in this tradition has involved the use of decision trees, although boosting techniques and rule learning with RIPPER have also been tried (Hoste, 2005; Ng and Cardie, 2002b, 2003a). For the system described here, I have employed various other machine learning methods, viz. memory-based learning, maximum entropy modelling, and support vector machines, with the main focus—here as elsewhere in this thesis—on the use of memory-based learning. It should be noted that memory-based learning was also one of the machine learning methods employed by Hoste (2005).

8.5.1 Filters

A number of filters are applied as a preprocessing step to filter out impossible antecedent candidates before the machine learning method is allowed to select among the remaining candidates. The filters implement the set of rules for gender and number match or mismatch given in this section. Note that when these rules proclaim a match, it does not necessarily mean that the candidate in question is selected as an

antecedent; it simply means that the candidate is not filtered out by the rules and thus goes on to be evaluated by the subsequent antecedent selection mechanism.

Gender

The following rules hold for gender match or mismatch between an anaphor and an antecedent candidate:

- If the candidate is unmarked for gender, it is always regarded as a match. This usually applies to proper names, because the tagger does not know the gender of most names. A notable exception is Norwegian first names, many of which are listed with gender in the tagger lexicon.
- If both the anaphor and the candidate are pronouns that are marked for gender (i.e., they are singular pronouns), then their genders must be identical. This means that, for instance, the only pronouns allowed to corefer with *hun* are the feminine-marked singular pronouns (*hun*, *henne*, and *hennes*) and the plural pronouns (*de*, *dem*, and *deres*), which are unmarked for gender.
- If the anaphor is a masculine pronoun and the candidate is a common noun which is marked with gender, it must be a masculine noun. The same logic applies when the anaphor is a feminine pronoun, except that in this case the candidate can be either feminine or masculine. The reason for this is that nouns denoting females often have masculine gender, while occurrences of the opposite case (i.e., feminine nouns that denote a male person) are much more rare (although exceptions exist, especially with derogative words such as *pyse* "sissy" and *skravlebøtte* "chatterbox"). I have verified empirically that allowing masculine pronouns to have feminine antecedents decreases the performance of the system, while allowing feminine pronouns to have masculine nouns as antecedents increases performance.

Number

The following rules hold for number match or mismatch between an anaphor and an antecedent candidate:

- If the candidate is not marked for number, it is always regarded as a match.
- If the candidate is a possessive pronoun, it is always regarded as a number match. The reason for this is that possessives functioning as antecedents agree in number with the head of their noun phrase rather than with the antecedent. For instance, in (1-a), *mine* and *Jeg* corefer, but *mine*, which is a plural possessive, agrees in number with the plural noun *bøkene* functioning as the head of its noun phrase, rather than with the singular pronoun *Jeg*.

Jeg and *mine* are first-person pronouns, which are not handled by the system. Although third-person possessives do not have a special plural form which distinguishes them from the singular, they are nevertheless marked with number by the tagger in agreement with the number of the head of the phrase, as shown in (1-b).

(1) a. Dette er bøkene $mine_i^{\text{PL}}$. Jeg_i^{SG} kjøpte dem i går.
this is books my I bought them yesterday
"These are my books. I bought them yesterday."

b. Dette er bøkene $hennes_i^{\text{PL}}$. Hun_i^{SG} kjøpte dem i går.
this is books her she bought them yesterday
"These are her books. She bought them yesterday."

Note that although a possessive is allowed to corefer with a pronominal anaphor regardless of its number, it still has to match the pronoun with regard to gender, as described in the previous section.

- If the candidate is a proper name, it is always regarded as a match. This is because with proper names (especially WORK names, such as names of books, movies, etc.) we cannot be sure of anything with respect to number. For instance, proper names that are apparently plural may nevertheless be referred to by a singular pronoun. This is equally true in English and Norwegian, as the following example illustrates:

(2) I watched ["The Bridges of Madison County"]$_i^{\text{PL}}$ yesterday. It$_i^{\text{SG}}$ was boring.

- If the candidate is the (rare) polite second-person pronoun *De*, it is always counted as a match. This is a singular pronoun, but it is incorrectly marked as plural by the Oslo-Bergen tagger, and hence we need to take special measures to make sure it is allowed to corefer with a singular anaphor (as well as a plural one, because plural anaphors may have several co-ordinated singular or plural antecedents).

- If the anaphor is singular and the candidate is unambiguously plural, it is a mismatch. Conversely, however, we cannot treat a plural anaphor and a singular antecedent as a mismatch because of the potential for plural anaphors to have several co-ordinated singular or plural antecedents.

8.5.2 Dealing with reflexive and possessive antecedents

Reflexive and possessive pronouns occurring as antecedents pose a particular challenge for the system, because they are not marked with the gender or number of their binders (i.e., their syntactically and semantically defined antecedents), and this leaves the system without any agreement clues as to whether they are suitable antecedent candidates for later anaphors. Consider (3) as an illustration.

(3) a. Da Mari$_i$ kom hjem, sto Petter$_j$ og barberte seg$_j$. Han$_j$ hadde
when Mari came home stood Petter and shaved REFL he had
dårlig tid.
bad time
"When Mari came home, Petter was shaving. He was in a hurry."

b. Da Mari$_i$ kom hjem, sto Petter$_j$ og barberte seg$_j$. Hun$_i$ hadde
when Mari came home stood Petter and shaved REFL she had
ikke tid til å vente.
not time to to wait
"When Mari came home, Petter was shaving. She did not have time to wait."

In its search for antecedents for *Han* in (3-a) and for *Hun* in (3-b), the system cannot determine whether *seg* is a suitable candidate just by looking at this word in isolation, since it does not in itself carry any information about the gender of its referent. Thus, in order to determine whether *seg* is a suitable antecedent candidate in each of these cases, the system needs access to information about the gender of the binder of *seg*. This is done by percolating feature values (including gender and number) from the binder to the reflexive.

In fact, it turns out that the usefulness of feature value percolation from antecedent to anaphor is not restricted to cases involving reflexives; it is valuable in all cases where a potential antecedent lacks information about a certain feature, but this information can be retrieved from earlier markables in the same coreference chain. I will return to this point in section 8.5.5.

8.5.3 Pronoun-specific classifiers

The coreference resolution system presented by Hoste (2005) consists of three different classifiers: one for pronouns, one for common nouns, and one for proper nouns. Hoste hypothesizes that this is beneficial because these different types of NPs rely on different kinds of information for their coreference resolution. For links between proper nouns, for example, string matching and aliasing (e.g., *IBM* vs. *International Business Machines*) is important, while the resolution of coreference between a pronoun and a common noun to a much larger extent relies on gender and number match along with the distance between them (Hoste, 2005, p. 37). Hoste (2005, p. 114) in

fact concludes that her results give no conclusive evidence that using separate classifiers is beneficial. Nevertheless, she does achieve significantly better performance with separate classifiers in three out of six experiments.

My own work deals with pronominal anaphors only. However, I have taken Hoste's approach a step further and created different classifiers for different types of pronouns: one for *han* "he" and *hun* "she", one for *den* "it (masc./fem.)", and one for *de* "they" (*det* "it (neut.)" is treated by special means).

While it seems intuitive that pronouns, common nouns, and proper nouns may benefit from different types of features, it is perhaps less clear that the distinction between different pronouns could also benefit from a separation of classifiers. There is reason to expect different behaviour between these pronouns, however. Based on linguistic intuition, we might expect *han* and *hun* to be likely to corefer with nouns denoting human beings and with nouns that function as the SUBJECT of the sentence, while *den* is expected to corefer with non-human nouns and more often with nouns that have other syntactic functions. *De* is special in that it often corefers with a group of NPs. Although coreference with a group cannot be handled by the system (which only links *de* to the closest member of the group), this special aspect of *de* is likely to make it benefit from specially tuned feature weights.

As it turns out, when the full AR system presented in this chapter is tested with one vs. three classifiers on the development corpus, resolution of *han/hun* and *de* is not affected significantly, but *den* experiences a significant increase in accuracy from 33.96% to 54.72% (significant at the 5% level; $p \leq 0.0266$). Furthermore, Table 8.3 on page 200 illustrates that the machine learning algorithm arrives at fairly different gain ratio weights for the three classifiers, meaning that the features have different degrees of importance for the various pronoun types.

8.5.4 Features

When using vector-based machine learning techniques, we need to define a set of features that will constitute the feature vectors. Most of the features I use for the AR task are boolean, each feature answering a particular question either about the antecedent or (in most cases) about the anaphor/antecedent pair.

When determining the set of features, I have used the work by Soon et al. (2001) as my starting point, because their system by far outperforms earlier machine learning systems on the MUC-6 data set (cf. section 7.2.6). However, not all of their features are relevant for pronominal anaphora resolution (Soon et al. do general coreference resolution), and some of their features do not perform very well on the Norwegian data and have therefore either been excluded or modified for use in my system.

Finally, I have introduced a few additional features, which have been shown by experiments to provide valuable information to the system.

Features inspired by Soon et al.

The following list shows which features have been carried over from Soon et al., either directly or in a modified form. The features are numbered in the same way as in the tables found in the following sections. Refer to section 7.2.6 for a more detailed description of Soon et al.'s features.

1. & 2. **Distance**: In Soon et al.'s system, the value of the **Distance** feature is the number of sentence boundaries between anaphor and antecedent. This turns out not to work very well in the Norwegian system. In experiments without feature percolation (cf. section 8.5.5), using this kind of feature with the default Overlap similarity metric (cf. section 2.2.1) led to a significant decrease in performance. A likely reason for this is the following. As much previous work has shown (e.g., Lappin and Leass (1994); Mitkov (1998, 2002); Holen (2006)), candidates that are close to the anaphor are more likely to be an antecedent than candidates that are located further away from it. However, with the Overlap metric, TiMBL will see the distance values simply as either identical or different, without any conception of the fact that the different values express different distances between anaphor and antecedent.

   Changing the similarity metric to Numeric for this particular feature improves the results somewhat. With a numeric similarity metric, the system is able to see that some distance values are closer than others, and since many anaphor–antecedent pairs in the training data will have small distances, smaller distances in the test data will also be preferred.

   Alternative ways of expressing similarity between feature values are to use the Modified Value Difference Metric (cf. section 2.2.3) or the Jeffrey Divergence Metric (cf. section 2.2.4). However, when applied to the **Distance** feature, MVDM produces only an intermediate accuracy, while the Jeffrey Divergence Metric performs even worse than Overlap. Hence, the Numeric metric is the best choice if this feature is to be used.

   On the other hand, there is a better solution to expressing a preference for small distances between anaphor and antecedent: we can replace Soon et al.'s single multi-valued distance feature with a set of two boolean features. The first feature indicates whether the antecedent is in the same sentence as the anaphor or in the previous sentence. The second feature indicates whether the antecedent is in the same, previous, or penultimate sentence.

   In this way, we express the degree of closeness between anaphor and antecedent in a way which is meaningful to TiMBL: the smallest distances (same or previous sentence) will turn on both features, an antecedent candidate in the penultimate sentence will only turn on feature number two, while antecedents that are even

further away will not turn on any of these features. Replacing the single distance feature with these two boolean features boosts yields the best accuracy on the development corpus.

3. **Lemma Match**: Soon et al. found string matching between the anaphor and the antecedent candidate to be the most important feature. This finding has later been confirmed by the work of Yang et al. (2004b) and Hoste (2005) (although, in Hoste's case, the predominance of this feature is moderated by feature selection and parameter optimization), and the feature also turns out to be the most important one for the Norwegian system.

   In Soon et al.'s work, this feature indicates whether there is a perfect match between the form of the anaphor NP and the form of the antecedent NP after any determiners have been removed (meaning, for instance, that *this house* will match *that house* or just *house*).

   I have found that a modified version of this feature works best. First of all, my system works with pronominal anaphors only and with antecedent candidate *nouns* rather than full *NP*s (in addition to antecedent pronouns, of course). Hence, there is no need to strip the anaphor and antecedent candidate of any determiners. Secondly, I am using lemmas rather than inflected forms; hence, in my system the feature is called Lemma Match rather than String Match. Thus, *han* will match either *han* (the SUBJECT form) or *ham* (the oblique form). Finally, the matching procedure allows either the anaphor or the antecedent to be a substring of the other; no full match is required.

   This last point is hardly relevant when only pronominal anaphors are considered. However, although it is not the focus of this thesis, the system is in fact equipped to do full coreference resolution (where anaphors can be nouns as well as pronouns), and preliminary tests have indicated that substring matching should be preferred over full string matching in order to obtain the best results.

4. **Gender Agreement**: This feature is included in the Norwegian system along with the Gender filter described in section 8.5.1. Unlike the case of number agreement (see below), with gender agreement it turns out that it pays off to use a combination of a hard filter and a machine learning feature.

   A difference between the feature implementations in Soon et al.'s system and in the Norwegian system is that, in the former system, the feature is assigned a value of "unknown" unless both the anaphor and the antecedent are marked with gender, while in my system, a lack of gender marking on either element leads to a feature match. This strategy was determined experimentally to be the most beneficial.

Additional features

In addition to those features that have been carried over from Soon et al. (2001), either directly or in a modified form, I have introduced a number of additional features. These include features that take advantage of the information provided by the named entity recognizer, PP attachment disambiguator, and animacy detection mechanism that are described in previous chapters. These support modules have resulted in four features that are presented towards the end of the list below.

5. **Reflexive and closest** SUBJECT: This feature represents an approximation to Principle A of Chomsky's Binding Theory (Chomsky, 1981, 1986). Principle A states that all reflexives and reciprocals must be bound in their domain. Binding is defined in terms of c-command:

   X c-commands Y iff:

   - X does not dominate Y
   - Y does not dominate X
   - The first (branching) node that dominates X also dominates Y

   In the terms of traditional grammar, this means that reflexives and reciprocals occurring as DIRECT OBJECTs (the most common syntactic function filled by such elements) are bound by the SUBJECT of the same sentence, and this is what the feature is meant to capture. In (4) for example, the reflexive DIRECT OBJECT *seg* "himself" is bound by the SUBJECT *David*:

   (4) David$_i$ barberte seg$_i$ i morges.
   David shaved REFL in morning
   "David shaved this morning."

   As shown in Table 8.3 on page 200, this feature receives one of the highest gain ratio weights in the memory-based learner, showing that it provides the system with some very valuable information. Somewhat surprisingly perhaps, implementing it as a machine learning feature works better than a "hard" constraint that always links a reflexive with the SUBJECT of the same sentence. Having it as a soft constraint makes it susceptible to being overridden by other features, and apparently this is beneficial for the system. Furthermore, constraining the feature to consider only the SUBJECT of the current sentence actually decreases performance; the best results are obtained by looking for the closest SUBJECT to the left without any regard for sentence boundaries.

   One type of situation where such a behaviour might be desirable consists of cases of erroneous grammatical tagging. Erroneous tagging may lead to a non-

sentence-boundary conjunction being tagged as a sentence boundary, or the actual SUBJECT of the sentence being tagged as some kind of non-nominal which cannot be classified as a SUBJECT. In both of these cases, the actual SUBJECT will be unavailable as a binder of the reflexive, and a different candidate might be selected instead. The other machine learning features will then have the opportunity to turn down this candidate on other grounds.

Another kind of phenomenon which might benefit from the use of soft constraints and disregard of sentence boundaries is the phenomenon of implicit binders. Hellan (1988) argues that reflexives may be bound by implicit arguments, as long as the implicit argument is part of a coreference chain which contains at least one visible argument. This is the case, for instance, in (5), where it is claimed that *bok* "book" has an implicit author argument that is able to bind *seg selv* because it corefers with *Jon*, which is a visible element (Hellan, 1988, p. 176):

(5) En bok om seg selv ville gi Jon stor selvtillit.
a book about REFL SELF would give Jon great self-confidence
"A book about himself would give Jon great self-confidence."

In (6), one may also postulate an implicit argument for *kritikk* "criticism". In this case, however, there is no visible element with which this implicit argument corefers, and Hellan therefore claims that the sentence is ungrammatical:

(6) Kritikk av sine lærere er straffbart.
criticism of REFL-POSS teachers is punishable
"Criticism of one's teachers is punishable."

Lødrup (2007), on the other hand, points out that utterances such as (6) are perfectly acceptable to many Norwegian speakers. He goes on to argue that, for such speakers, reflexives may be bound by implicit arguments that are not linked to any visible elements. Furthermore, he points out that some young speakers even allow reflexive forms with a generic interpretation to occur without any binder at all (Lødrup, 2007, p. 13):

(7) En motorsag kan (...) skade seg selv og andre.
a chain-saw can hurt REFL SELF and others
"A chain saw can hurt oneself and others."

Although I have not established to what extent such constructions occur in my corpus, the existence of implicit binders and even binder-less reflexives may be

part of the reason why a soft constraint works better than a hard one. Since the system only considers overt elements (i.e., words), it cannot choose an implicit binder as the antecedent of a reflexive. By not forcing the system to select the SUBJECT of the sentence, we allow it to search further towards the beginning of the text for the closest visible antecedent (note that, since the system does not handle cataphora, it will not consider the cataphor *Jon* when resolving the reflexive in (5)). In those cases where no visible antecedent can be found, the system has the possibility of leaving the reflexive without an antecedent.

6. **Non-reflexive and SUBJECT of same sentence**: This feature is similar to the previous one, but approximates Principle B of the Binding Theory instead. This principle states that (other) pronouns (i.e., not reflexives or reciprocals) should *not* be bound in their domain. For example, the DIRECT OBJECT *ham* "him" in (8) cannot be bound by the SUBJECT *David*:

   (8) David$_i$ møtte ham$_j$ i går.
   David met him yesterday
   "David met him yesterday."

   Unlike the previous feature, for this feature to be useful, it has to be restricted to antecedent candidates that are the SUBJECT of the same sentence as the one in which the anaphor occurs. With this restriction, the feature boosts the performance of the system (otherwise, it decreases it). Compared to the previous feature, however, the gain ratio weight of this feature is very small, indicating that it is not terribly important for the performance of the system.

7. **Syntactic function of antecedent**: This feature indicates whether the syntactic function of the antecedent candidate is SUBJECT, OBJECT, or something else (represented by the value *other_func*).

   Some form of SUBJECT preference is commonly included in AR systems, regardless of underlying methodology. For example, Lappin and Leass (1994) have a factor called *subject emphasis*, and the same function is filled by the *First noun phrases/Givenness* indicator of Mitkov (1998) and the *Obliqueness* indicator of Mitkov (2002). Likewise, many machine learning approaches include a feature which contains information about whether an antecedent candidate is a SUBJECT, such as Ng and Cardie (2002b), Fisher, Soderland, McCarthy, Feng, and Lehnert (1995), Strube et al. (2002), and Hoste (2005), as well as the machine learning system presented in this chapter. In centering-based systems which order centers by grammatical function (which most such systems do), SUBJECTs are always placed at the top of the function hierarchy (cf. section 7.2.4). Furthermore, not only do many systems include some form of SUBJECT preference;

it has also proved to be an important factor; for instance, the *subject emphasis* factor of Lappin and Leass' system receives a high score.

Interestingly, however, Holen (2006) does not include a SUBJECT preference factor, because it actually decreases the performance of her system, and I have confirmed that this is the case for my reimplementation of her system as well. Holen argues that this may be caused by the widespread use of Norwegian presentational constructions, in which the SUBJECT role is filled by a pleonastic *det* "it". Nevertheless, for the machine learning system, the **Syntactic function of antecedent** feature does prove to be valuable, as is evident in the relatively strong weight listed for this feature in Table 8.3.

Judging from Holen's findings and the importance of this feature for the machine learner, one might at first be tempted to hypothesize a preference for functions *other* than SUBJECT in the system. As it turns out, however, this is not the case; there are other reasons for the successful use of this feature.

First, the present system employs a gender filter, both for the machine learning approach (cf. section 8.5.1) and for my reimplementation of Holen's system (cf. section 8.10.2). This filter ensures that pleonastic pronouns (which are neuter) cannot be selected as antecedents of those singular anaphors that are handled by the system (except for *det* itself, of course, but this pronoun is always assumed to be pleonastic and is left out of most evaluations reported in this chapter; cf. section 8.7 for details). Thus, presentational constructions can only be a potential problem for the plural pronoun *de*, reducing the impact of such constructions considerably.

A second reason for the value of this feature can be found in the difference between the weights that the feature receives in the classifiers for *han/hun*, *den*, and *de*, respectively (cf. Table 8.3 on page 200). In the training corpus for the *han/hun* classifier, 68.5% of the positive examples have a SUBJECT antecedent, while only 29.0% of the negative examples have one. In other words, for *han/hun*, this feature does indeed function as a SUBJECT preference feature, and an important one at that, as its gain ratio weight shows. For the *den* classifier, 33.3% of the positive examples and 35.3% of the negative ones have a SUBJECT antecedent, ruling out any SUBJECT preference for this classifier and resulting in a very low gain ratio weight for the feature. Finally, for the *de* classifier, 56.6% of the positive examples and 32.5% or the negative ones have a SUBJECT antecedent. Thus, this classifier should have a certain preference for SUBJECT antecedent, but a weaker one than the *han/hun* classifier. Accordingly, its gain ratio weight lies somewhere between the weights found for the other classifiers.

Interestingly, these facts demonstrate the importance of the decision made in this work to use different classifiers for different anaphors. By letting the machine learning method work out separate weights for this feature depending on the type of anaphor, we allow the system to make use of the syntactic function of the antecedent only in those cases where it is beneficial. Furthermore, we allow it to weight the feature differently depending on *how good* a clue this information provides for a given anaphor type.

8. **Syntactic parallelism**: This feature indicates whether the anaphor and the antecedent have the same syntactic function. Information about syntactic parallelism has been included in many earlier AR systems, within Centering Theory (e.g., Kameyama, 1986), in indicator/factor-based systems (e.g., Lappin and Leass, 1994; Mitkov, 2002), and in systems based on machine learning (e.g., Hoste, 2005; Strube et al., 2002).

9. **Concatenated lemmas**: Unlike most of the other features, this is not a binary feature. Rather, it consists of a concatenation of the anaphor lemma and the antecedent candidate lemma. Although this would seem to make the system too closely tuned to the training corpus, it does in fact prove to be a rather important feature, getting a fairly high gain ratio weight (cf. Table 8.3) and leading to a marked performance improvement on the development and test corpora as well, especially for cases where the antecedent is not identical to the anaphor.

10. **Antecedent in animate noun list (filtered)**: This is a binary feature which indicates whether the noun is found in one of the animate noun lists extracted using the offline approach described in chapter 6. Two versions of this feature were tested: one using the list obtained by filtered queries and one using the list obtained by unfiltered queries (cf. chapter 6 for details). The filtered version performed best and was therefore selected to be the one included in the final feature set.

    A feature based on the manually obtained noun lists used by Holen (2006) was also tried. However, such a feature degraded the performance of the system, regardless of whether it was used as an alternative to the **Antecedent in animate noun list** feature or as a supplement to it. This might seem surprising at first, especially since Holen's lists have been compiled manually and hence are expected to have a very high precision. However, as was pointed out in section 6.4.2 with respect to the automatically extracted noun lists, the AR system performs best when the lists are as large as possible, even if this happens at the expense of precision. Hence, a plausible explanation is that the restricted size of Holen's lists makes them generate too many false negatives (i.e., cases where a noun is in fact animate but is not found in any of the lists).

As an alternative to using Holen's lists in a separate feature, I have also tested the effect of combining her lists with the automatically compiled lists and using them in the same feature. However, it turns out that the development corpus contains only five nouns that are found in Holen's lists but not in the lists extracted from the Web (*demonstrant* "demonstrator", *familie* "family", *forelder* "parent", *passasjer* "passenger", and *tysker* "German"), and they have no impact on the performance of the system.

11. **Online animacy evaluation**: This is a binary feature whose value is the result of submitting the candidate to the (pseudo-)online approach to animacy detection described in chapter 6.

12. **Antecedent is a person name**: This binary feature indicates whether the candidate is classified as a person by the named entity recognizer described in chapter 4.

13. **Antecedent is embedded**: This feature indicates whether the candidate occurs as the head of a PP which is attached to a noun, thus making it embedded in the NP that has the other noun as its head. The attachment site is determined by the PP attachment disambiguator presented in chapter 5.

    Recall that the disambiguator only considers sequences of verbs followed by a noun and a PP. Furthermore, with regard to embeddedness, we are only interested in the noun that occurs as the head of the PP. If the candidate does not occur in such a position, this feature is assigned the value *N/A* (not applicable). If, on the other hand, the candidate does occur in such a position, the feature is assigned the value *N* if the disambiguator decides on noun attachment, and *V* if it selects verb attachment.

The resulting feature set

Table 8.3 sums up the preceding discussion of various features by listing those features that turned out to be beneficial and hence were included in the final system. The only exception is the **Antecedent is embedded** feature, which is not beneficial to system performance but is still included in the table because it is studied in particular depth through my development of the PP attachment disambiguator described in chapter 5. The table also shows the gain ratio weight (cf. section 2.2.2) that each feature was assigned in the classifiers for the different pronoun types.

Note that since the morphological and syntactic information that is used by some of these features is taken from an automatically annotated corpus, they will not always be correct. Hence, unlike some of the earlier work in this field (e.g., Hobbs, 1978;

Feature #	Feature name	*han/hun*	*den*	*de*
1	Distance ≤ 1	1.42	1.78	2.56
2	Distance ≤ 2	1.18	0.46	1.83
3	Lemma match	**34.70**	**23.71**	**26.29**
4	Gender agreement	5.05	**11.39**	0
5	Reflexive and closest SUBJECT	**28.33**	0	0
6	Non-reflexive and SUBJECT of same sentence	1.35	1.08	0.13
7	Syntactic function of antecedent	6.45	0.03	1.92
8	Syntactic parallelism	2.23	1.28	1.18
9	Concatenated lemmas	8.70	**15.38**	6.90
10	Antecedent in animate noun list (filtered)	**10.06**	**16.82**	0.99
11	Online animacy evaluation	2.21	0.12	1.29
12	Antecedent is a person name	2.17	1.32	0.84
13	Antecedent is embedded	3.35	6.50	1.40

Table 8.3: Features used in the TiMBL-based AR system, along with their gain ratio weights multiplied by 100. Weights above 10 are shown in bold. The feature numbers refer to the numbering used in the feature descriptions in the text.

Tetreault, 2001), the present system is evaluated on data that have been analysed using fully automatic means. Information about animacy and embeddedness is also acquired automatically, using the techniques described in earlier chapters.

The performance of the individual features will be discussed thoroughly in section 8.7.

Features from Soon et al. that were left out

Since I used the feature set of Soon et al. (2001) as a starting point for my own features, a few words about those features that were left out are in order. The excluded features were the following:

- ***i*-Pronoun**: I am only dealing with pronominal anaphora, so the anaphor is always a pronoun. Hence, this feature is redundant and consequently excluded.
- ***j*-Pronoun**: This feature indicates whether the antecedent candidate is a pronoun (personal, reflexive, or possessive). Including this feature actually decreases the performance of the system; hence, it is left out. It is not clear why performance deteriorates when this feature is included. However, in a system that only deals with pronominal anaphors, a large part of the function of this feature is probably assumed by the Lemma Match feature and the Concatenated Lemmas feature together. It might be more useful in a system that deals with general co-occurrence (as indeed it was in Soon et al.'s system).
- **Definite Noun Phrase**: In the original system, this feature is true iff the antecedent candidate is a definite noun phrase. In addition to the indication

that the candidate is something different from a pronoun or a demonstrative, one might expect information about definiteness to be valuable, since definite NPs are more likely to express known information and hence to refer to entities that are more established in the discourse. Such entities are more likely to be referred to again (cf. the *Lexical Reiteration* indicator of Mitkov (1998, 2002) and the *Frequent Candidates* indicator of Mitkov (2002)).

In my own system, however, a feature which indicates whether the antecedent candidate is a definite noun actually decreases the performance. Hence, somewhat surprisingly, this kind of information turns out to be detrimental rather than valuable for the Norwegian system. Admittedly, as was the case with the j-Pronoun feature, some of the benefit of such a feature with regard to frequency of occurrence is probably provided by the Concatenated Lemmas feature. However, the latter feature reflects the frequency of the candidate in the training data rather than in the test data, and it is therefore not expected to reflect the centrality of the entity in the test text.

- **Demonstrative Noun Phrase**: This feature parallels the j-Pronoun feature and the Definite Noun Phrase feature in Soon et al.'s system, and, like the other two features, having such a feature decreases the performance of the Norwegian system.

- **Semantic Class Agreement**: In Soon et al.'s system, semantic classes are looked up in WordNet. Since no WordNet exists for Norwegian, no corresponding feature could be included in my system. However, arguably the most important aspect of semantic classes in connection with AR is whether or not the anaphor and antecedent candidate agree with respect to animacy, and features that check for animacy agreement are included in the system, as explained earlier in this section.

- **Number Agreement**: In the Norwegian system, this feature is replaced by a filter (cf. section 8.5.1). Experiments confirm that using such a "hard" filter works better than using a "soft" machine learning feature. It should be pointed out, however, that the filter does not simply remove candidates that do not have the same number marking as the anaphor; the procedure is a bit more involved, as the description in section 8.5.1 shows.

- **Both-Proper-Names**: Since only pronominal anaphors are considered in the Norwegian system, this feature is irrelevant.

- **Alias**: This was one of the most important features in Soon et al. (2001)'s system. As the Norwegian system only deals with pronominal anaphors, however, the feature is also irrelevant there.

- **Appositive**: This feature, which indicates whether the candidate is in apposition to the anaphor, was also found to be an important feature by Soon et al., but again it is fairly irrelevant for the Norwegian system, for two reasons. First, the system focuses on pronominal anaphora, and although one might find cases of appositions to pronouns, they are relatively rare. An example is given in (9).

  (9) Han, den rikeste mannen i bygda, skulle ikke la seg kommandere.
  "He, the richest man of the village, would not let himself be told what to do."

  More importantly, however, these constructions are not relevant for a system which only handles anaphora and not cataphora, since the apposition will always occur after the pronoun (note that it is defined by Soon et al. as indicating apposition to the cataphoric element itself, not the general property of being appositive) and will therefore not be considered as a potential antecedent.

Features in later work that have been left out

Various work on machine learning approaches to AR appearing after the Soon et al. (2001) paper has introduced a number of additional features to this task. Hoste (2005) gives an overview of the different types of features used in the machine learning AR literature and selects a set of 41 features, most of them taken from the literature, but also some novel ones. Hoste's set of 41 features is much larger than the 13 features used in the present thesis. However, those of Hoste's features that have been excluded have all been left out for good reason, because they are either irrelevant, unachievable given the current state of Norwegian language resources, or detrimental to the performance of the classifier. The following list provides more detail about the reasons for excluding each of these features.

- *Positional features*: Hoste's DIST_SENT (sentence distance) and DIST_NP (NP distance) features both lower the performance of the classifier regardless of whether they are treated as numeric or symbolic, and regardless of the distance metric applied. The DIST_LT_THREE (distance less than three sentences) feature, on the other hand, has proven useful and has been extended by a "distance less than two" feature in the Norwegian system (cf. the discussion of the *Distance* feature above).

- *Local context features*: These are 12 features which represent the word forms and the parts-of-speech of three words to the left and three words to the right of the anaphor. Applying these features to the Norwegian data decreases performance from 74.60% to 63.94%. If we keep only those features that were

found by Hoste to be most informative for pronominal anaphors (LEFT_WD_3, LEFT_WD_2, LEFT_WD_1, LEFT_POS_3, LEFT_POS_2, RIGHT_POS_1, and RIGHT_POS_2), performance reaches the slightly better, but still inferior level of 65.36%. An attempt to replace word forms by lemmas is even more devastating, bringing the performance down to 43.34%. Hence, these kinds of features have been left out of the Norwegian system.

It is not clear why these features would be detrimental for the present system. On the other hand, it is hard to see how they could be beneficial for the kinds of pronouns that are considered in this thesis. The most plausible effect of such features would be to distinguish between referential and pleonastic pronouns, in that the local context of the pronoun has been shown to provide some clues as to the referentiality of the pronoun (Boyd, Gegg-Harrison, and Byron, 2005; Denber, 1998; Evans, 2001). Since the only potentially pleonastic pronoun *det* "it" is not handled by the Norwegian system (except in a few experiments where it is always assumed to be pleonastic; cf. section 8.7), the system will not benefit from any such effects.

- *Morphological and lexical features*: I_PRON (anaphor is a pronoun), J_PRON (antecedent candidate is a pronoun), and I+J_PRON (both are pronouns) were left out because the information that the anaphor is a pronoun is redundant, while the information that the antecedent is a pronoun actually decreases the performance of the system (as discussed above).

  J_PRON_I_PROPER (anaphor is a pronoun and antecedent candidate is a proper name) was tested but made no difference to the performance of the system. This is probably due to the fact that the proper name information is already partly encoded by the *Antecedent is a person name* feature, and the anaphor is always a pronoun. This hypothesis is supported by the fact that if the *Antecedent is a person name* feature is removed, the J_PRON_I_PROPER feature makes a big difference (72.11% with the feature vs. 68.20% without it).

  J_DEMON (antecedent is a demonstrative) and J_DEF (antecedent is a definite NP) were detrimental to the performance of the system, as discussed above.

  I_PROPER (anaphor is a proper name) and BOTH_PROPER (both anaphor and antecedent are proper names) are not applicable in a system that only deals with pronominal anaphora, while J_PROPER (antecedent is a proper name) appears to be sufficiently covered by the *Antecedent is a person name* feature, as stated above.

  NUM_AGREE is replaced by a filter. As mentioned earlier, a "hard" filter performs better than a machine learning feature for this property.

- *Syntactic features*: Hoste's system includes the following syntactic features: ANA_SYN (the syntactic function of the anaphor, represented as SUBJECT, OBJECT, or something else), ANT_SYN (same thing for the antecedent candidate, except that if the candidate is a SUBJECT or an OBJECT, the value also indicates whether it is the closest one to the anaphor), BOTH_SBJ/OBJ (whether both anaphor and antecedent are SUBJECTs or OBJECTs, i.e., syntactic parallelism), and APPOSITION (whether the anaphor is in apposition to the antecedent candidate).

  The Norwegian system includes an equivalent to the ANT_SYN feature, except that it does not indicate whether a SUBJECT or OBJECT is the closest one to the anaphor because this turns out to damage performance. An equivalent to the BOTH_SBJ/OBJ feature is also included (i.e., the *Syntactic parallelism* feature). The ANA_SYN feature, on the other hand, is not included, since it lowers the performance of the system. The APPOSITION feature is also left out, because pronouns are rarely found in appositive position. Note that Hoste's APPOSITION feature is different than the one used by Soon et al. (2001), since the latter indicates whether the antecedent candidate is in apposition to the anaphor rather than the other way round, as discussed above.

- *String-matching features*: Since the Norwegian system only deals with pronominal anaphors, there is no need for features like Hoste's PART_MATCH and ALIAS features. A single string-matching feature is sufficient; this feature indicates whether the anaphor and antecedent candidate lemmas match, as explained above.

- *Semantic features*: Hoste's semantic features make use of named entity recognition, gender information that can be deduced from the form of the NP (such as the use of *Mr.* or *Mrs.*), and information from WordNet (Fellbaum, 1998) for English and EuroWordNet[2] and CELEX (Baayen, Piepenbrock, and van Rijn, 1993) for Dutch.

  No WordNet or EuroWordNet exists for Norwegian, and hence features such as SYNONYM, HYPERNYM, or Hoste's various other features that refer to semantic classes cannot be used with the Norwegian system. However, gender agreement is handled partly by the gender filter and partly by the *Gender agreement* feature, and the named entity recognizer and the animacy detection procedures described in earlier chapters provide the system with valuable semantic information, as will be shown in the results section.

[2] `http://www.illc.uva.nl/EuroWordNet/`

8.5.5 Feature percolation

The way that the AR task is formulated in this thesis, there is a focus on finding the closest antecedent for each anaphor. There are some cases, however, where the closest antecedent candidate lacks some important information that can nevertheless be retrieved by examining any earlier antecedents which the candidate has already been linked to. Therefore, if an antecedent is part of an already established coreference chain, most of its features are also allowed to match on earlier markables in the chain. This is called *feature percolation*. The feature percolation procedure is reminiscent of Yang et al. (2004b)'s technique of letting the linking of an anaphor be evaluated against an already established chain of coreferential NPs, each of which may provide some important information that could influence the linking decision.

Feature #	Feature	Percolates?
1	Distance ≤ 1	No
2	Distance ≤ 2	No
3	Lemma match	Yes
4	Gender agreement	Yes
5	Reflexive and closest SUBJECT	No
6	Non-reflexive and SUBJECT of same sentence	No
7	Syntactic function of antecedent	No
8	Syntactic parallelism	No
9	Concatenated lemmas	No
10	Antecedent in animate noun list (filtered)	Yes
11	Online animacy evaluation	Yes
12	Antecedent is a person name	Yes
13	Antecedent is embedded	No

Table 8.4: Percolation or non-percolation of feature values. The feature numbers refer to the numbering used in the feature descriptions in section 8.5.4.

Table 8.4 shows which of the features are allowed to percolate and which are not. In section 8.7, performances with and without feature percolation are compared for the TiMBL runs, showing that feature percolation does in fact have a positive effect. Note that percolation requires the ability to dynamically create feature vectors during testing, since the percolating feature values come from those markables that the system has already determined to be part of the coreference chain. In other words, the system's own decisions influence the feature vectors that it is presented with later on.

This means that we cannot apply the normal procedure of preparing the whole feature vector set beforehand and then presenting the entire set to the machine learning algorithm. Rather, we need the ability to present the vectors to the machine learner one by one. With TiMBL, this can easily be achieved by running it as a server which will load the instance base and then sit and wait, listening on a certain computer port

for classification requests. No doubt similar mechanisms could be implemented for the MaxEnt and SVM algorithms by feeding them a series of test files containing only a single test vector that had been created on the fly based on the earlier classifications made by the algorithm. However, due to the natural way that feature percolation can be implemented with TiMBL, and because it turns out that TiMBL performs as well as or better than the other algorithms on the development corpus when percolation is switched off (as seen in section 8.7), feature percolation has only been tested with TiMBL.

8.6 Training and testing procedure

As mentioned in section 8.5.4, training and testing vectors are created by pairing an anaphor with an antecedent candidate. The overall training and testing procedure is inspired by Soon et al. (2001). I create positive training examples by pairing each anaphor with its actual antecedent. Negative examples are created by pairing the anaphor with each of the markables that occur between the anaphor and the actual antecedent.

During the testing phase, the system starts by pairing an anaphor with its closest preceding markable and asks the machine learner to classify the vector as either a positive or a negative example of an anaphor–antecedent pair. If it is classified as negative, the system continues with the next antecedent candidate, and so on until it either finds a vector that gets a positive classification or it reaches a certain predefined distance from the anaphor. In the present experiments, the threshold distance is set to be 20 markables; this was experimentally established to be a good threshold value for the BREDT data (the best multiple of 10).

If no candidate is classified as an antecedent, the system looks for a "backup antecedent", which is the closest candidate that gets through the filters and matches on the **Gender agreement** feature as well as at least one of the features that look for a match in animacy, i.e., **Antecedent in animate noun list (filtered)**, **Online animacy evaluation**, and **Antecedent is a person name**. If still no suitable candidate is found, the anaphor is said to lack an antecedent.

8.7 Results

8.7.1 Testing on the development corpus

Default parameter settings

When reporting on classifier performance directly in this section, I employ the commonly used measures of recall, precision, and F-score for the classes of positive and negative examples, as well as the accuracy of the classifier as a whole.

When reporting on the complete anaphora resolution procedure, on the other hand, I only use accuracy. This is because I am focusing on the ability of the system to find correct antecedents (i.e., the class of positive examples); I am not particularly concerned about its ability to find non-antecedents. Thus, I consider each of the relevant pronouns in the development corpus in turn, and check whether the system is able to find the correct antecedent for those pronouns that have an antecedent, or to refrain from selecting an antecedent for those that do not (the latter cases would not be captured if performance had been expressed in terms of F-score on the positive and negative classes). The results of this testing procedure are appropriately represented by accuracy. Furthermore, by using accuracy, I can compare my results directly to those of Holen (2006), who uses the same measure.

Tables 8.5 to 8.7 show classifier performance on the development corpus using default parameter settings, while Table 8.8 on page 208 and Table 8.9 on page 210 display results for the entire anaphora resolution procedure.

	han/hun	*den*	*de*	ALL
SVM	93.46	76.47	82.47	89.47
MaxEnt	94.53	58.82	80.83	88.35
TiMBL	92.95	84.87	78.60	88.91

Table 8.5: Classification accuracy on the development corpus with default parameter settings. The figures are for classification of both antecedents and non-antecedents prior to clustering.

	han/hun	*den*
SVM	77.93 / 95.84 / 85.96	0 / 0 / 0
MaxEnt	83.10 / 95.0 / 88.65	91.07 / 35.42 / 51.00
TiMBL	81.71 / 89.93 / 85.63	80.36 / 64.29 / 71.43
	de	ALL
SVM	48.33 / 50.88 / 49.57	66.27 / 86.04 / 74.88
MaxEnt	51.67 / 46.62 / 49.01	78.20 / 74.06 / 76.07
TiMBL	58.33 / 42.68 / 49.30	77.47 / 76.12 / 76.79

Table 8.6: Recall/precision/F-score on the development corpus for positive examples, i.e., actual anaphor–antecedent pairs, using default parameter settings.

	han/hun	*den*
SVM	98.83 / 92.83 / 95.74	100.00 / 76.47 / 86.67
MaxEnt	98.49 / 94.40 / 96.40	48.90 / 94.68 / 64.49
TiMBL	96.84 / 93.87 / 95.33	86.26 / 93.45 / 89.71
	de	ALL
SVM	89.87 / 88.91 / 89.39	96.67 / 90.23 / 93.34
MaxEnt	87.16 / 89.26 / 88.20	91.50 / 93.12 / 92.30
TiMBL	83.00 / 90.18 / 86.44	92.46 / 92.97 / 92.72

Table 8.7: Recall/precision/F-score on the development corpus for negative examples, i.e., pronouns paired with non-antecedents, using default parameter settings.

Focusing on the performance on all pronouns given in the last column of Table 8.5, the support vector machines perform significantly better than the MaxEnt models when only classifier accuracy is considered (the differences between SVM and TiMBL and between TiMBL and MaxEnt in the same column are not statistically significant). However, when we inspect the performance of the SVMs separately on positive (antecedent) and negative (non-antecedent) classifications given in Table 8.6 and Table 8.7, we see that the classifier for *den* is in fact not able to identify a single antecedent; its relatively high classification accuracy on *den* is entirely due to the fact that it classifies everything as a non-antecedent. Consequently, it obtains a very low performance on this pronoun in the full AR system, as shown in Table 8.8. Note that the reason why it obtains a non-zero accuracy on *den* in the full system despite having zero recall on antecedents for this pronoun is that it correctly refrains from selecting an antecedent in those cases where none should be selected.

	han/hun	*den*	*de*
SVM	66.27	13.21	41.98
MaxEnt	69.02	49.06	39.69
TiMBL	72.75	52.83	44.27
TiMBL w/percolation	75.10	54.72	46.56
TiMBL w/percolation + backup	76.67	54.72	46.56

Table 8.8: Full AR accuracy on the development corpus for the different anaphors by the various machine learning methods using default parameter settings.

Table 8.8 shows how well each machine learning method performs on each of the anaphor types, and it also displays the effect of feature percolation (cf. section 8.5.5) and the use of a backup antecedent (cf. section 8.6). Both feature percolation and backup antecedents yield significant improvements on *han/hun* (backup antecedents at the 5% level: $p \leq 0.0215$; percolation at the 1% level: $p \leq 0.00183$), but neither of them affects resolution of the other pronouns significantly (backup antecedents actually have no impact on the other pronouns at all).

The accuracy figures displayed in Table 8.8 are considerably lower than the F-

scores obtained on positive anaphor–antecedent examples in the pure classification experiments. This is mainly caused by the fact that the full AR procedure has a much lower precision on the positive class than the isolated classification stage does. This is in turn a natural consequence of the different ways that antecedent candidates are presented to the system in these experiments.

In the pure classification experiment, the test data are created in the same way as the training data. This means that the only negative examples presented to the system are those in which the anaphor is paired with a non-antecedent located *between* the actual antecedent and the anaphor itself. In other words, even if the classifier does not find the correct antecedent, there are relatively few pairs that can be mistakenly classified as positive for each anaphor. In the full anaphora resolution case, on the other hand, we keep presenting the system with feature vectors until it either classifies one as a positive example or we reach the threshold distance of 20 markables. Thus, if the system does not detect the correct antecedent, it has plenty of opportunity to pick a wrong one.

Of course, in the pure classification experiment, the classifier has the opportunity to wrongly classify *several* non-antecedents as antecedents for a single anaphor, while the full AR system only selects one antecedent per anaphor. Nevertheless, this turns out to be less damaging to the precision than the opportunity to select among 20 candidates.

Additionally, in some cases, the full AR system chooses a candidate which is closer to the anaphor than the actual antecedent. Such cases not only lower the precision because of the wrong classification (as they do in the pure classification experiments as well), but they are also detrimental to recall because, unlike the classifier in a pure classification experiment, the system never gets the chance to select the real antecedent.

On the other hand, the full AR system benefits from the filters described in section 8.5.1, which are not applied in the pure classification experiments. The clearest effect of filtering is seen with TiMBL, which achieves the highest scores in the full system even on pronouns on which its classifier score is actually lower than those of the other algorithms (i.e., *han/hun* and *de*). TiMBL was the machine learning algorithm used during the development of the system, and the filters seem to have been implemented in a way which is especially beneficial for this particular machine learning algorithm.

Table 8.9 shows the result of evaluating the full AR system on various combinations of pronouns. Without feature percolation and backup antecedents, TiMBL performs significantly better than both MaxEnt and SVM when *de* is included in the test set; when it is not included, the difference between TiMBL and MaxEnt is not statistically

	han/hun/den	*han/hun/den/de*
SVM	61.28	57.64
MaxEnt	67.14	61.96
TiMBL	70.87	65.85
TiMBL w/percolation	73.18	68.16
TiMBL w/percolation + backup	74.60	69.31
	han/hun/den/de/det	*han/hun/den/det*
SVM	66.53	70.21
MaxEnt	69.51	73.97
TiMBL	72.19	76.37
TiMBL w/percolation	73.78	77.85
TiMBL w/percolation + backup	74.58	78.77

Table 8.9: Full AR accuracy on the development corpus for combinations of anaphors by the various machine learning methods using default parameter settings.

significant. Adding feature percolation and backup antecedents creates a system that significantly outperforms all of the others.

The set of pronouns evaluated by Holen does not include the plural pronoun *de*[3]. In a sense this is reasonable, since her system is not able to select more than one antecedent for an anaphor, or to select a conjunct of two or more NPs as its antecedent, even though both of these situations often occur with *de*. The same is the case with the system presented in the current work; only a single word can be selected as the antecedent of a given anaphor. When evaluated on the same set of anaphors (*han*, *hun*, and *den*) that was used in Holen (2006)'s original evaluation, the TiMBL-based system reaches an accuracy of 74.60%.

In addition to the evaluation on *han/hun/den*, Table 8.9 shows the results of evaluating a set of anaphors that includes the *de* pronoun; in this case, the accuracy drops to 69.31%. On the other hand, if resolution of the neuter singular pronoun *det* is added, performance increases considerably, to an accuracy of 74.58% with *de* included and 78.77% without it. However, the high performance on *det* resolution is due to the fact that it is always assumed to be pleonastic, which is actually the case in about three quarters of all occurrences in the test data. Although this simple solution works fairly well due to the skewed distribution of referential and pleonastic occurrences of *det*, a more satisfactory approach would involve trying to separate the two types of occurrences. This has to be left for future work, however (cf. section 8.12.2).

Optimized parameter settings

As in the previous chapters, I have used Paramsearch to optimize machine learning parameters. The values found by Paramsearch are shown in Tables 8.10 to 8.12.

[3]As mentioned in chapter 7, this is not explicitly stated in the text itself, but has been established through personal communication with the author.

	Dist. metric	Metric threshold	Weight. scheme	k	Extrapol. method
han/hun	MVDM	2	No weights	7	IL
den	Overlap	N/A	Gain ratio	1	N/A
de	MVDM	1	Information gain	5	ID

Table 8.10: Optimal settings found by Paramsearch for the TiMBL classifiers.

	Parameter estimation	Gaussian prior
han/hun	L-BFGS	0
den	GIS	0
de	L-BFGS	0

Table 8.11: Optimal settings found by Paramsearch for the MaxEnt classifiers.

	Training error/margin trade-off	Cost factor	Kernel function	d/γ
han/hun	10	0.5	Polynomial	1
den	50	0.5	Radial basis	0.016
de	1000	2.0	Radial basis	0.002

Table 8.12: Optimal settings found by Paramsearch for the SVM classifiers. The last column shows the d parameter (the exponent) for the polynomial function or the gamma parameter for the radial basis function.

	han/hun	*den*	*de*
SVM	93.60	76.43	88.27
MaxEnt	93.36	73.57	88.08
TiMBL	93.13	75.00	89.22

Table 8.13: Classification accuracy on the training corpus with optimized parameter settings, as reported by Paramsearch.

Table 8.13 shows the accuracy figures reported by Paramsearch on the training corpus for the different pronoun-specific classifiers. Note that Paramsearch can only work on the classification stage. Hence, these figures are for classification only and do not take into account the subsequent clustering stage. The figures are accuracy levels for both positive and negative examples, i.e., both antecedents and non-antecedents.

	han/hun	*den*	*de*	ALL
SVM	94.07	84.45	79.35	89.82
MaxEnt	94.53	85.71	80.98	90.62
TiMBL	91.77	84.87	80.39	88.53

Table 8.14: Classification accuracy on the development corpus with optimized parameter settings. The figures are for classification of both antecedents and non-antecedents prior to clustering.

When the classifiers are applied to the development corpus using the optimized parameter values, we obtain the accuracy figures in Table 8.14. The table reports on the accuracy obtained separately by each classifier as well as the overall accuracy on the entire development corpus. Again, the figures are for the classification stage only.

With optimized parameters, MaxEnt performs better than the other machine learning methods on the set of all pronouns, although the difference is significant only at the 5% level (MaxEnt vs. TiMBL: $p \leq 0.0115$; MaxEnt vs. SVM: $p \leq 0.0140$). SVM and MaxEnt perform significantly better than TiMBL on *han/hun*, while TiMBL and MaxEnt perform significantly better than SVM on *de*. The differences on *den* are not statistically significant.

There is no significant difference between TiMBL's overall classification accuracy using default and optimized parameters. MaxEnt retains its accuracy on *han/hun* and shows a non-significant difference on *de*, but its improvement on *den* is large enough to cause a significant overall increase in accuracy ($p \ll 0.001$). The performance of the support vector machine for *han/hun* is not significantly affected by optimization, while the SVM for *den* improves its accuracy significantly ($p \leq 0.00108$). On the other hand, SVM performance on *de* actually drops significantly ($p \leq 0.00646$), and the overall performance of SVM is not changed significantly by parameter optimization.

Tables 8.15 and 8.16 show recall, precision, and F-score on antecedents and non-antecedents, respectively. As was the case with default parameters, all classifiers perform much better on non-antecedents than on antecedents. This is hardly surprising, considering the overwhelming majority of negative examples in the training corpus. For each positive anaphor–antecedent example, there is one example for every non-antecedent that occurs between the anaphor and the antecedent (cf. section 8.6), leading to a training corpus that consists of 36.2% positive and 63.8% negative examples (if reflexives are excluded, the distribution becomes even more skewed, with

30.9% positive and 69.1% negative examples, since reflexives tend to occur close to their antecedents).

	han/hun	*den*
SVM	82.50 / 93.68 / 87.74	50.00 / 75.68 / 60.22
MaxEnt	87.67 / 90.74 / 89.18	50.00 / 82.35 / 62.22
TiMBL	74.55 / 91.91 / 82.33	80.36 / 64.29 / 71.43
	de	ALL
SVM	54.17 / 43.62 / 48.33	74.82 / 80.76 / 77.68
MaxEnt	54.17 / 47.10 / 50.39	78.65 / 81.16 / 79.88
TiMBL	55.83 / 45.89 / 50.38	71.72 / 78.04 / 74.75

Table 8.15: Recall/precision/F-score on the development corpus for positive examples, i.e., actual anaphor–antecedent pairs, using optimized parameter settings.

	han/hun	*den*
SVM	98.07 / 94.19 / 96.09	95.05 / 86.07 / 90.34
MaxEnt	96.91 / 95.79 / 96.34	96.70 / 86.27 / 91.19
TiMBL	97.73 / 91.74 / 94.64	86.26 / 93.45 / 89.71
	de	ALL
SVM	84.81 / 89.50 / 87.09	94.47 / 92.36 / 93.41
MaxEnt	86.80 / 89.72 / 88.24	94.34 / 93.44 / 93.88
TiMBL	85.71 / 89.94 / 87.78	93.74 / 91.44 / 92.58

Table 8.16: Recall/precision/F-score on the development corpus for negative examples, i.e., pronouns paired with non-antecedents, using optimized parameter settings.

Finally, Table 8.17 shows the performance on each pronoun-specific classifier trained with optimized parameters, while Table 8.18 shows the accuracy on identifying antecedents obtained by the full system running classification and clustering.

	han/hun	*den*	*de*
SVM	70.20	39.62	40.46
MaxEnt	73.14	32.08	38.17
TiMBL	69.02	52.83	41.22
TiMBL w/percolation	70.59	54.72	43.51
TiMBL w/percolation + backup	74.90	54.72	43.51

Table 8.17: Full AR accuracy on the development corpus for the different anaphors by the various machine learning methods using optimized parameter settings.

Comparing Tables 8.9 and 8.18, which show performance on various pronoun combinations, and disregarding the effects of feature percolation and backup antecedents, we see that TiMBL with default parameter settings seems to obtain the highest overall accuracy, but it is in fact not significantly higher than MaxEnt with optimized parameters. It is, however, significantly higher than the performance of SVM. All the

	han/hun/den	*han/hun/den/de*
SVM	67.32	62.25
MaxEnt	69.27	63.40
TiMBL	67.50	62.53
TiMBL w/percolation	69.09	65.46
TiMBL w/percolation + backup	73.00	68.55
	han/hun/den/de/det	*han/hun/den/det*
SVM	69.71	74.09
MaxEnt	70.51	75.34
TiMBL	69.91	74.20
TiMBL w/percolation	71.10	75.23
TiMBL w/percolation + backup	73.29	77.74

Table 8.18: Full AR accuracy on the development corpus for combinations of anaphors by the various machine learning methods using optimized parameter settings.

remaining tests on the development corpus described in this chapter have been carried out using TiMBL with default parameter settings.

A likely reason for the good performance of TiMBL with default settings is that this is the setup that was used during the feature and filter selection process, meaning that the features and filters were chosen in a way that optimizes performance with that particular setup. This illustrates the importance of feature selection, and suggests that joint feature selection and parameter optimization along the lines of Hoste (2005) could be a promising direction for future development of the AR system.

8.7.2 Cross-validation results

The figures reported in the previous section were the results of training the machine learning system on the training part of the corpus and then testing it on the same material that has also been used for testing the Centering Theory and factor-based systems, as described in later sections. However, a common technique for maximizing the training and test material when using statistical methods is to run a cross-validation experiment, and such an experiment has therefore been carried out with the memory-based AR system. Using 10-fold cross-validation and evaluating performance on *han*, *hun*, and *den* with default parameter settings, the system reaches an accuracy of 70.58%. Although this is considerably lower than the 74.60% accuracy it achieves on the development corpus, it is still significantly better than the results obtained by the Centering Theory and factor/indicator-based approaches described later in this chapter when cross-validation is simulated by applying them to the entire corpus (cf. section 8.13 for further details).

Feature #	Feature	Accuracy
1	Distance ≤ 1	1.78
2	Distance ≤ 2	1.78
3	Lemma match	53.11
4	Gender agreement	1.78
5	Reflexive and closest SUBJECT	1.78
6	Non-reflexive and SUBJECT of same sentence	1.78
7	Syntactic function of antecedent	1.78
8	Syntactic parallelism	1.78
9	Concatenated lemmas	64.30
10	Antecedent in animate noun list (filtered)	49.02
11	Online animacy evaluation	1.78
12	Antecedent is a person name	1.78
13	Antecedent is embedded	1.78

Table 8.19: The performance of features in isolation on the development corpus. The feature numbers refer to the numbering used in the feature descriptions in section 8.5.4.

8.7.3 Testing features in isolation

Table 8.19 shows how each feature performs by itself when evaluated on the *han*, *hun*, and *den* pronouns. Interestingly, most of the features do not have any effect at all when used in isolation: they do not cause the classifier to select any antecedents at all, and the accuracy of 1.78% obtained with these features simply corresponds to the proportion of pronouns that do not in fact have an antecedent[4]. Thus, the tendency of the system to classify a candidate as non-antecedent is so strong that a single feature is rarely able to overcome it. This is hardly surprising, given the overwhelming majority of negative examples in the training data (cf. Hoste, 2005, chapter 7, for further discussion).

It should also be noted that two of the three features that do yield increased performance in isolation are string-matching features, a fact which is in accordance with the findings of Soon et al. (2001) (cf. section 7.2.6). The exception is the *Antecedent in animate noun list* feature. However, the good effect of this feature is no doubt due to the fact that the pronouns *han*, *hun*, and *den* are present in the animate noun lists, having been mistakenly extracted as animate (in the case of the first two pronouns) or inanimate (in the case of the last one) nouns by the offline extraction method described in chapter 6. Thus, in effect, this feature assumes most of the function of the *Lemma match* feature when the latter feature is switched off.

[4]Although only 74.0% of the total number of pronouns handled by the system have antecedents (cf. Table 8.1), the overwhelming majority of those that do not have an antecedent are occurrences of *det*, which are not handled by the machine learning algorithms.

8.7.4 Removing single features

In the previous section, each of the features was added to the classifier in turn and tested in isolation. Although this tells us something about the usefulness of each feature, we saw that the abundance of negative training examples allows only the most powerful features to have an effect in isolation. However, we can also approach the question of feature importance from the opposite angle, by starting with a full feature set and removing one feature at a time. This gives us a more finely tuned view of the influence of each feature, as we can see from Table 8.20.

Note that the figures listed in the table are obtained without using backup antecedents (cf. section 8.6). This is because the calculation of a backup antecedent in itself makes use of some of the features, and the influence of these features would therefore be counted twice if the backup procedure was used in this evaluation.

Feature #	Removed feature	*han/hun*	*den*	*han/hun/den*
	None	74.51	52.83	72.47
1	Distance ≤ 1	74.90	43.40	71.94
2	Distance ≤ 2	74.51	52.83	72.47
3	Lemma match	72.94	52.83	71.05
4	Gender agreement	73.92	41.51	70.87
5	Reflexive and closest SUBJECT	74.31	52.83	72.29
6	Non-reflexive and SUBJECT of same sent.	73.53	52.83	71.58
7	Syntactic function of antecedent	73.73	52.83	71.76
8	Syntactic parallelism	74.12	50.94	71.94
9	Concatenated lemmas	66.67	52.83	65.36
10	Antecedent in animate noun list (filtered)	73.92	43.40	71.05
11	Online animacy evaluation	74.31	52.83	72.29
12	Antecedent is a person name	70.00	50.94	68.20
13	Antecedent is embedded	75.10	54.72	**73.18**

Table 8.20: The performance of the classifier on the development corpus when single features are removed. The feature numbers refer to the numbering used in the feature descriptions in section 8.5.4. The highest accuracy is shown in bold.

There are several interesting points to note in this table. First, among the three features that caused an effect when used in isolation, only the *Concatenated lemmas* feature has a major impact when it is the only feature removed from the classifier. The effect of removing either the *Lemma match* feature or the *Antecedent in animate noun list* feature is non-significant and in fact smaller than the effect of removing *Gender agreement* (however, as explained in the next section, in the optimal system without the *Antecedent is embedded* feature, removing the *Antecedent in animate noun list* feature does have a significant impact).

Another thing to note is that half of the features (6 out of 13) only affect performance on *han/hun*. The goal of the feature selection process has been to maximize

overall performance, and because of the overwhelming majority of *han* and *hun* vs. other third-person pronouns, features that benefit resolution of these two pronouns have the biggest impact on performance and therefore have the greatest chance of being admitted into the system.

The *Distance* $\leq$ *2* feature only has an effect in combination with the use of backup antecedents, and hence it does not show any impact in Table 8.20.

Finally, it turns out that the *Antecedent is embedded* feature actually seems to decrease performance slightly on both *han/hun* and *den*; however, the decrease is not statistically significant. It is nevertheless somewhat surprising, since Table 8.3 showed that the gain ratio weights of this feature are actually higher than those of many of the beneficial features. Since the gain ratio weights reflect the degree to which a feature decreases the entropy of the instance base (cf. section 2.2.2), it is an indicator of the contribution that the feature makes to the predictability of class membership (antecedent vs. non-antecedent) in the training corpus. With relatively high gain ratio weights, and with the training and development corpora containing material from the same genre, we would also expect the *Antecedent is embedded* feature to make good predictions on the development corpus, but, as it turns out, this is not the case. Although the negative effect is not significant in itself, the feature is nevertheless removed from the classifiers that are used to produce the results reported elsewhere in this chapter.

8.7.5 The effect of the support modules

The support modules developed in the previous chapters provide several of the features that were selected for inclusion in the AR system: the named entity recognizer of chapter 4 provides values for the *Antecedent is a person name* feature, the PP attachment disambiguator described in chapter 5 contributes information for the *Antecedent is embedded* feature, while the animacy detection procedures from chapter 6 provide the system with information used by the *Antecedent in animate noun list (filtered)* and *Online animacy evaluation* features.

Removal of the *Antecedent is a person name* feature leads to the second most severe decrease in performance of the system, and the decrease is seen with both the *han/hun* classifier and the *den* classifier. The fact that this feature would have a positive impact was to be expected from its gain ratio weights (cf. Table 8.3), although the weights were not high enough to predict such a dramatic and highly significant effect ($p \ll 0.01$).

The gain ratio weights of the *Antecedent in animate noun list (filtered)* feature are quite high, both for the *han/hun* classifier and for the *den* classifier. For the resolution of *den*, it also has a huge impact on classifier accuracy, bringing it down to 43.40% when it is removed. Its effect on *han/hun* resolution is also positive, though much

less dramatic. In the optimal system (without the *Antecedent is embedded* feature), removal of the *Antecedent in animate noun list* feature has a significant impact on performance (at the 5% level; $p \leq 0.0201$). Removal of the *Online animacy evaluation* feature, on the other hand, does not have a statistically significant effect. Hence, of the two animacy detection procedures described in chapter 6, only the offline approach gives a statistically significant boost to the system.

The lack of a significant effect of the online evaluation might be due to the fact that online evaluation is restricted to a small set of templates and fillers in order to make the experiment practically feasible (cf. section 6.5). Future access to increased computing power may make it feasible to extend the number of patterns used, perhaps to the point where the full set of templates and fillers can be used and the effect of the online approach can be more appropriately compared to the offline one.

However, an equally important factor may be the fact that Google's frequency estimates seem rather unreliable[5]. For example, a search for *han er sjef* yields a frequency estimate of 5460; however, only 710 snippets are actually returned. Similarly, the query *hun er ansatt som forsker* results in a frequency estimate of 213, but only 125 snippets are returned. It could be that the estimates are correct but that, for some reason, only a subset of the matches are returned. However, since the frequency figures are estimates rather than accurate counts, it seems more likely that they are wrong. As the estimates form the foundation of the online evaluation approach, any unreliability in the estimates weakens the approach.

As discussed in the previous section, the *Antecedent is embedded* feature causes a non-significant decrease in performance, despite the fact that it substantially reduces the entropy in the training database. Hence, the experiments conducted in this work do not support the finding by Lappin and Leass (1994) that embeddedness makes a markable become less accessible as an antecedent—at least not to the extent that embeddedness can be determined by a state-of-the-art PP attachment disambiguator.

8.7.6 Testing on a separate test corpus

The previous sections describe how the different machine learning methods perform on the development corpus as well as in cross-validation. On the development corpus, the memory-based approach with feature percolation and backup antecedents significantly outperforms the alternative approaches that are not based on machine learning. Furthermore, the cross-validation results show that the significantly better performance of the memory-based approach carries over to parts of the corpus that were not used for evaluating the system during parameter optimization and general fine-tuning.

[5] A colleague of mine, Helge Lødrup, made me aware that the difference between the estimate and the snippets returned may be extreme in some cases: a search for *kjøkkenet* "the kitchen" gives an estimate of 595,000, while only 668 snippets are returned. The discrepancies I have noticed in my own work have been more modest, however.

Testing on the development corpus gives us an opportunity to make a fair comparison with the results reported by Holen (2006), since her test material was drawn from the same corpus (albeit an earlier version). However, in order to get a truly independent evaluation of the different approaches, I have created a test corpus that was not used during the development of the system at all (in fact, it was created after the development process was finished) and that consists of material that is not contained in the BREDT corpus. In addition to providing an opportunity for an independent evaluation, the use of a separate test corpus makes it possible to use the entire original corpus for training the machine learning methods.

The test corpus consists of small samples from 16 different novels, extracted from the beginning of each novel so as to make sure that no antecedents are missing from the samples. The corpus consists of 14,577 tokens and contains 1167 occurrences of the pronouns *han*, *hun*, *den*, *det*, *de*, and reflexives, all of which have been annotated with their closest antecedent (if any). Table 8.21 provides further details about the number of pronouns of the different types in the test corpus. The following sections describe how the various machine learning methods perform on the test corpus using either default or automatically optimized parameter settings.

han/hun/REFL	*den*	*det*	*de*	Total
758	36	274	99	1167

Table 8.21: The number of pronouns of the relevant types in the test corpus.

Default parameter settings

I start by reporting the results of using the default parameter settings of each machine learning algorithm when testing on the test corpus. The tables in this section, as well as in the next section, which deals with optimized parameter settings, parallel those that were shown in section 8.7.1 for the development corpus.

Overall classifier performance on each pronoun type is shown in Table 8.22, with separate classifier performance on the positive (antecedent) and negative (non-antecedent) classes shown in Tables 8.23 and 8.24, respectively.

Table 8.25 displays the results of the entire AR system (performing both classification and clustering) on each pronoun type, while Table 8.26 shows the performance on combinations of pronouns.

As we see in Table 8.26, maximum entropy modelling outperforms SVM on all tested pronoun combinations. When *de* is excluded, the difference between the algorithms is only significant at the 5% level and bordering on significance at the 1% level ($p \leq 0.0161$ on *han/hun/den*), while when *de* is included, the difference is significant even at the 1% level ($p \leq 0.00388$ on *han/hun/den/de*).

	han/hun	*den*	*de*	ALL
SVM	92.71	62.34	72.99	90.38
MaxEnt	92.61	76.62	73.72	90.68
TiMBL	91.47	76.62	72.99	89.60

Table 8.22: Classification accuracy on the test corpus with default parameter settings. The figures are for classification of both antecedents and non-antecedents prior to clustering.

	han/hun	*den*
SVM	74.33 / 95.70 / 83.67	9.38 / 100.0 / 17.14
MaxEnt	76.34 / 93.00 / 83.85	53.13 / 85.00 / 65.38
TiMBL	73.40 / 90.89 / 81.21	62.50 / 76.92 / 68.97
	de	ALL
SVM	22.83 / 87.50 / 36.21	66.51 / 95.39 / 78.38
MaxEnt	25.00 / 88.46 / 38.98	70.07 / 92.58 / 79.77
TiMBL	36.96 / 68.00 / 47.89	66.86 / 89.14 / 76.41

Table 8.23: Recall/precision/F-score on the test corpus for positive examples, i.e., actual anaphor–antecedent pairs, using default parameter settings.

	han/hun	*den*
SVM	98.88 / 91.98 / 95.31	100.0 / 60.81 / 75.63
MaxEnt	98.07 / 92.51 / 95.21	93.33 / 73.68 / 82.35
TiMBL	97.53 / 91.61 / 94.48	86.67 / 76.47 / 81.25
	de	ALL
SVM	98.35 / 71.60 / 82.87	98.86 / 89.26 / 93.82
MaxEnt	98.35 / 72.18 / 83.26	98.00 / 90.21 / 93.95
TiMBL	91.21 / 74.11 / 81.77	97.11 / 89.19 / 92.98

Table 8.24: Recall/precision/F-score on the test corpus for negative examples, i.e., pronouns paired with non-antecedents, using default parameter settings.

	han/hun	*den*	*de*
SVM	66.09	13.89	26.26
MaxEnt	66.89	36.11	31.31
TiMBL	65.30	52.78	31.31
TiMBL w/percolation	73.09	52.78	32.32
TiMBL w/percolation + backup	73.35	52.78	32.32

Table 8.25: Full AR accuracy on the test corpus for the different anaphors by the various machine learning methods using default parameter settings.

	han/hun/den	*han/hun/den/de*
SVM	63.73	59.57
MaxEnt	65.49	61.70
TiMBL	64.74	61.03
TiMBL w/percolation	72.17	67.75
TiMBL w/percolation + backup	72.42	67.97

	han/hun/den/de/det	*han/hun/den/det*
SVM	67.61	71.44
MaxEnt	69.24	72.75
TiMBL	68.72	72.19
TiMBL w/percolation	73.86	77.72
TiMBL w/percolation + backup	74.04	77.90

Table 8.26: Full AR accuracy on the test corpus for combinations of anaphors by the various machine learning methods using default parameter settings.

TiMBL performs worse than SVM on *han/hun*, but better on *den* and *de*, and the overall difference between the two machine learning algorithms on the pronoun combinations shown in Table 8.26 is not significant.

The differences between TiMBL and MaxEnt on the various pronoun combinations are also not significant. In fact, their classification accuracy on *den* is identical (cf. Table 8.22), although MaxEnt is a little better at finding non-antecedents for this pronoun, while TiMBL is best at finding antecedents, as shown in Tables 8.23 and 8.24. TiMBL's superior ability to find antecedents enables it to achieve a much better result for *den* in the full AR system, as shown in Table 8.25. However, the better performance by MaxEnt on the much more frequent *han/hun* pronouns levels out the overall difference between the algorithms, and with performance on *de* again being identical there is no significant difference between the two on any of the pronoun combinations shown in Table 8.26.

The use of percolation with TiMBL yields a huge boost in performance for all pronoun combinations ($p \ll 0.01$), while the effect of using a backup antecedent is non-significant on this corpus.

Optimized parameter settings

When Paramsearch is run on the entire BREDT corpus, it arrives at the optimal parameter settings shown in Tables 8.27 to 8.29. Table 8.30 lists the accuracy figures obtained during optimization as they are reported by Paramsearch.

Table 8.31 shows the classifier accuracy for each pronoun-specific classifier on the test corpus, while Tables 8.32 and 8.33 list information about classifier performance on the positive and negative classes separately. Finally, Tables 8.34 and 8.35 show the results for the entire anaphora resolution system.

	Dist. metric	Metric threshold	Weight. scheme	k	Extrapol. method
han/hun	MVDM	2	No weights	3	ID
den	Jeffrey	2	No weights	11	ED1
de	Jeffrey	2	Gain ratio	3	ED1

Table 8.27: Optimal settings found by Paramsearch for the TiMBL classifiers on the entire BREDT corpus.

	Parameter estimation	Gaussian prior
han/hun	L-BFGS	100
den	L-BFGS	1.4142
de	GIS	10

Table 8.28: Optimal settings found by Paramsearch for the MaxEnt classifiers on the entire BREDT corpus.

	Training error/margin trade-off	Cost factor	Kernel function	d/γ
han/hun	1000	0.5	Radial basis	0.128
den	100	1.0	Polynomial	3
de	1000	0.5	Radial basis	0.032

Table 8.29: Optimal settings found by Paramsearch for the SVM classifiers on the entire BREDT corpus. The last column shows the d parameter (the exponent) for the polynomial function or the gamma parameter for the radial basis function.

	han/hun	*den*	*de*
SVM	93.76	86.18	88.44
MaxEnt	93.89	84.54	86.71
TiMBL	92.87	85.25	88.44

Table 8.30: Classification accuracy on the entire BREDT corpus with optimized parameter settings, as reported by Paramsearch.

	han/hun	*den*	*de*	ALL
SVM	92.44	70.13	72.99	90.32
MaxEnt	93.04	72.73	70.44	90.71
TiMBL	91.33	67.53	73.72	89.33

Table 8.31: Classification accuracy on the test corpus with optimized parameter settings. The figures are for classification of both antecedents and non-antecedents prior to clustering.

	han/hun	*den*
SVM	76.20 / 92.38 / 83.52	34.38 / 84.62 / 48.89
MaxEnt	81.68 / 89.72 / 85.51	37.50 / 92.31 / 53.33
TiMBL	72.86 / 90.83 / 80.86	25.00 / 88.89 / 39.02

	de	ALL
SVM	28.26 / 76.47 / 41.27	69.61 / 91.42 / 79.04
MaxEnt	14.13 / 86.67 / 24.30	72.94 / 89.70 / 80.46
TiMBL	25.00 / 88.46 / 38.98	66.06 / 90.71 / 76.44

Table 8.32: Recall/precision/F-score on the test corpus for positive examples, i.e., actual anaphor–antecedent pairs, using optimized parameter settings.

	han/hun	*den*
SVM	97.89 / 92.45 / 95.09	95.56 / 67.19 / 78.90
MaxEnt	96.86 / 94.03 / 95.42	97.78 / 68.75 / 80.73
TiMBL	97.53 / 91.46 / 94.40	97.78 / 64.71 / 77.88
	de	ALL
SVM	95.60 / 72.50 / 82.46	97.68 / 90.05 / 93.71
MaxEnt	98.90 / 69.50 / 81.63	97.03 / 90.99 / 93.91
TiMBL	98.35 / 72.18 / 83.26	97.60 / 89.00 / 93.10

Table 8.33: Recall/precision/F-score on the test corpus for negative examples, i.e., pronouns paired with non-antecedents, using optimized parameter settings.

	han/hun	*den*	*de*
SVM	65.70	41.67	29.29
MaxEnt	66.36	33.33	9.09
TiMBL	64.38	11.11	27.27
TiMBL w/percolation	72.43	27.78	29.29
TiMBL w/percolation + backup	72.43	27.78	29.29

Table 8.34: Full AR accuracy on the test corpus for the different anaphors by the various machine learning methods using optimized parameter settings.

	han/hun/den	*han/hun/den/de*
SVM	64.61	60.69
MaxEnt	64.86	58.68
TiMBL	61.96	58.12
TiMBL w/percolation	70.40	65.85
TiMBL w/percolation + backup	70.40	65.85
	han/hun/den/de/det	*han/hun/den/det*
SVM	68.47	72.10
MaxEnt	66.92	72.28
TiMBL	66.50	70.13
TiMBL w/percolation	72.41	76.40
TiMBL w/percolation + backup	72.41	76.40

Table 8.35: Full AR accuracy on the test corpus for combinations of anaphors by the various machine learning methods using optimized parameter settings.

With respect to pure classifier accuracy using default or optimized parameter settings, as shown in Tables 8.22 and 8.31, respectively, SVM does not exhibit any significant differences for any of the pronoun-specific classifiers (despite the seemingly large difference in accuracy score for the *den* pronoun, the relatively small number of occurrences of this pronoun makes the difference non-significant). For MaxEnt, a significant difference is found for the *de* classifier; this difference is significant at the 5% level and bordering on significance at the 1% level ($p \leq 0.0117$). However, neither overall classification performance nor performance on *han/hun* or *de* exhibit any significant differences. Finally, no significant differences are found for TiMBL (again, despite a seemingly large difference on the *den* classifier).

Thus, we may conclude that parameter optimization seems to have minimal impact on the pure classification accuracy for this task: on the development corpus, only MaxEnt shows a significant overall improvement after parameter optimization (cf. section 8.7.1), while on the test corpus, parameter optimization does not have any effect on overall classification accuracy for any of the machine learning methods.

Parameter optimization does, however, have an impact on the ability of the various machine learning algorithms to handle positive vs. negative classes, although in most cases this results in a decrease in performance for the full AR process. As shown in Tables 8.23 and 8.32, TiMBL's F-score on antecedents for the *den* pronoun drops from 68.97 to 39.02 after parameter optimization, and this in turn causes its accuracy on full resolution of *den* to drop from 52.78% to 11.11%. Combined with a decrease from 65.30% to 64.38% on *han/hun* and from 31.31% to 27.27% on *de*, this yields highly significant decreases in performance on all of the pronoun combinations shown in Table 8.35 ($p \ll 0.01$ in all cases). For MaxEnt, the accuracy on *de* drops from 31.31% to 9.09% after optimization, yielding a significant performance decrease on the pronoun combinations that include *de* ($p \ll 0.01$).

For SVM, on the other hand, the effect of parameter optimization is positive: although the difference between SVM with default settings and SVM with optimized settings does not emerge as statistically significant in itself, the improvement brought about by parameter optimization means that SVM reaches a performance level that is not significantly different from that of MaxEnt and TiMBL when these methods are used with default parameters. Moreover, when all methods are run with optimized parameter settings, SVM outperforms TiMBL on all pronoun combinations and MaxEnt on combinations that involve *de*, although in both cases the differences are significant only at the 5% level.

To summarize, all machine learning methods exhibit a lower AR performance on the test corpus than on the development corpus, as is to be expected. MaxEnt and TiMBL perform better than SVM in most cases, i.e., on the development corpus using

either default or optimized parameters and on the test corpus using default parameters. When the algorithm parameters are optimized on the entire BREDT corpus, however, SVM outperforms TiMBL, and in some cases MaxEnt, on the test corpus, although this is largely due to the fact that both TiMBL and MaxEnt experience significant performance decreases as a consequence of parameter optimization.

As we will see in section 8.10.3, my reimplementation of Holen (2006)'s ARN system also exhibits a lower performance on the test corpus than on the development corpus. This could be an indication that the test corpus just happens to be more difficult to handle for any feature-based approach. Alternatively, it could be caused by the fact that the development corpus is part of the BREDT corpus, and the BREDT corpus was also used by Holen in her development of ARN (albeit in an earlier version). Whatever the reason, the full TiMBL-based system with default parameter settings, percolation, and backup antecedents performs much better than the ARN reimplementation on the test corpus, with the difference being significant at the 1% level ($p \ll 0.01$). As was the case with the development corpus, the TiMBL-based system also significantly outperforms the Centering Theory-based system on the test corpus (cf. section 8.9.3).

The effect of individual features and support modules

Feature #	Feature	Accuracy
1	Distance ≤ 1	1.89
2	Distance ≤ 2	1.89
3	Lemma match	53.33
4	Gender agreement	1.89
5	Reflexive and closest SUBJECT	1.89
6	Non-reflexive and SUBJECT of same sentence	1.89
7	Syntactic function of antecedent	1.89
8	Syntactic parallelism	1.89
9	Concatenated lemmas	66.04
10	Antecedent in animate noun list (filtered)	51.45
11	Online animacy evaluation	1.89
12	Antecedent is a person name	1.89
13	Antecedent is embedded	1.89

Table 8.36: The performance of features in isolation on the test corpus. The feature numbers refer to the numbering used in the feature descriptions in section 8.5.4.

Table 8.36 shows the accuracy scores obtained when only single features are used, while Table 8.37 displays the effect of removing single features. The tests are run using the memory-based version of the system with feature percolation but without backup antecedents, as explained in section 8.7.4.

As was the case with the development corpus, most features do not have an ef-

Feature #	Removed feature	*han/hun*	*den*	*han/hun/den*
	None	73.22	47.22	72.04
1	Distance ≤ 1	73.12	47.22	71.95
2	Distance ≤ 2	73.52	52.78	**72.58**
3	Lemma match	72.20	61.11	71.70
4	Gender agreement	72.20	25.00	70.06
5	Reflexive and closest SUBJECT	72.33	52.78	71.45
6	Non-reflexive and SUBJECT of same sent.	72.46	41.67	71.07
7	Syntactic function of antecedent	71.54	38.89	70.06
8	Syntactic parallelism	72.73	50.00	71.70
9	Concatenated lemmas	66.80	52.78	66.16
10	Antecedent in animate noun list (filtered)	73.65	33.33	71.82
11	Online animacy evaluation	73.25	44.44	71.94
12	Antecedent is a person name	69.17	52.78	68.43
13	Antecedent is embedded	73.12	52.78	72.17

Table 8.37: The performance of the classifier on the test corpus when single features are removed. The feature numbers refer to the numbering used in the feature descriptions in section 8.5.4. The highest accuracy is shown in bold.

fect in isolation, yielding only a small accuracy score that is obtained by refraining from selecting any antecedent in those cases where none should be selected. Those that do have an effect are the same as those that have an effect on the development corpus: two string-matching features (*Lemma match* and *Concatenated lemmas*) and the *Antecedent in animate noun list* feature, which presumably takes over most of the function of the *Lemma match* feature when the latter is left out, as discussed in section 8.7.4.

The results in Table 8.37 are very similar to those in Table 8.20 of section 8.7.4. The features that cause the biggest performance drops when they are left out are *Concatenated lemmas* and *Antecedent is a person name*. They are followed by *Gender agreement*, but unlike on the development corpus, the *Syntactic function of antecedent* feature has an equally strong effect as the *Gender agreement* feature. Unlike on the development corpus, removal of the *Distance* $\leq$ *2* feature seems to yield a higher accuracy score, but the effect is not statistically significant.

With respect to the support modules described in earlier chapters, the *Antecedent is embedded* feature does not have any more significant effect on the test corpus than it has on the development corpus. The named entity information encoded in the *Antecedent is a person name* feature, on the other hand, has a highly significant positive effect, as it also did on the development corpus ($p \ll 0.01$).

Both of the animacy features (*Antecedent in animate noun list* and *Online animacy evaluation*) seem to cause small accuracy decreases when they are removed from the system, but none of the decreases reach the level of statistical significance, unlike on

the development corpus, where removing the *Antecedent in animate noun list* feature had a significant effect. As mentioned above, however, the latter feature does have a strong and highly significant effect on the test corpus performance when it is used in isolation.

8.8 Problems with the Soon et al. antecedent selection mechanism

The technique used by Soon et al. (2001) to select an antecedent for an anaphor is called a *close-first clustering technique* by Ng (2005). Using a close-first technique, the system starts at the first markable preceding the anaphor and moves towards the beginning of the text, stopping when it finds an antecedent candidate that the machine learner classifies as an antecedent.

Although this seems like a convenient way of implementing a preference for close antecedents, there are serious problems with this technique. First of all, it ignores a fact which is acknowledged in most other approaches to anaphora resolution, viz. the preference for SUBJECT antecedents.

For example, in my system, I prefer candidates that have the same syntactic function as the anaphor, i.e., candidates that match on the *Syntactic parallelism* feature. Hence, if the anaphor is a SUBJECT, and the previous sentence contains a SUBJECT and an OBJECT, both matching the anaphor on features such as gender and number, I want to select the SUBJECT candidate due to its syntactic parallelism with the anaphor. Furthermore, for *han* and *hun*, the *Syntactic function of antecedent* feature takes the role of a SUBJECT preference feature, as discussed in section 8.5.4.

Salience-based approaches, such as those of Lappin and Leass (1994) and Kennedy and Boguraev (1996), also make SUBJECTs likely to be chosen as antecedents by giving them a high salience value, in accordance with both linguistic theory (Keenan and Comrie, 1977) and experimental results.

A similar preference for SUBJECT antecedents is seen in centering approaches. This is most evident in those approaches that rely on grammatical ranking, such as Grosz et al. (1983, 1995), Brennan et al. (1987), and Tetreault (2001). However, those variants that use information structure for ranking the Cf list (Strube and Hahn (1996, 1999), as well as Strube (1998)'s "alternative to centering") also exhibit an indirect SUBJECT preference, since in SVO languages the SUBJECT most often realizes the leftmost hearer-old discourse entity.

Given this established preference for SUBJECT antecedents, the Soon et al. clustering technique runs into problems in cases where the SUBJECT and the OBJECT are equally good antecedent candidates according to other criteria (gender and number

match, lemma match, etc.). Assuming a canonical SVO word order, the OBJECT will be closer to the anaphor than the SUBJECT, and thus it will be chosen as the antecedent, even if the SUBJECT would have been preferable.

An alternative antecedent selection mechanism which avoids this trap is the so-called *best-first clustering* used by Aone and Bennett (1995), Ng and Cardie (2002b), and Iida, Inui, Takamura, and Matsumoto (2003). Rather than simply selecting the closest possible antecedent, best-first clustering selects the closest antecedent *with the highest coreference likelihood value*. Surprisingly, however, Ng (2005) demonstrated worse results for this selection mechanism than for Soon et al.'s mechanism on various types of news texts. Although Ng does not discuss possible reasons for this result, it shows that it is not necessarily beneficial to let classifier confidence override close-first selection, possibly because the error margin of the classifier is too large to make its confidence a reliable clue.

The Soon et al. approach exhibits an additional weakness, caused by the combination of the close-first technique with a more general trait of classification approaches to anaphora resolution, viz. that each antecedent candidate is considered in isolation. In other words, the actual classification of a markable as an antecedent or a non-antecedent does not in any way depend on any properties of the other antecedent candidates, since the vector to be classified only encodes information about the anaphor and the specific candidate under consideration.

As discussed above, if the previous utterance contains both a SUBJECT and an OBJECT which are suitable antecedent candidates, we prefer the SUBJECT. However, if the SUBJECT turns out *not* to be a suitable candidate (perhaps because it is removed by one of the filters or causes a mismatch on one or more strong features), then we want to consider the OBJECT instead.

The problem with using a close-first technique, then, is that, at the time we consider the OBJECT, we have not yet looked at the SUBJECT, so we do not know whether it is a suitable candidate. Also, because each candidate is classified in isolation, no information about the SUBJECT is allowed to influence the classification of the OBJECT. In other words, our assessment of the OBJECT as the preferred antecedent candidate depends on the suitability of the SUBJECT candidate, but that information is not available to the classifier at the time we ask it to classify the OBJECT.

8.8.1 A suggested alternative: Using Cf ranking to cluster classified instances

A natural solution to the problems with the close-first clustering approach described above would be to combine the machine learning classification approach with the kind of antecedent candidate ranking which is found in centering approaches, described there as Cf ranking. As an alternative to the simplistic and somewhat ad-hoc close-

first technique (and to the best-first technique which was shown by Ng (2005) not to work very well), the available literature on centering provides a number of alternative Cf ranking methods that are built on linguistic theory as well as psycholinguistic evidence.

Thus, by combining the strengths of machine learning classification with the linguistically founded Cf ranking methods, we should be able to produce a system that combines the best of two worlds. In fact, since the machine learning approach takes the feature idea from the factor/indicator approaches and applies machine learning to the weights on those features, a system which additionally borrows its clustering technique from Centering Theory would unify all of the most widely used types of approaches to anaphora resolution.

A clustering technique that is based on Cf ranking has been tested on the development corpus used in previous sections. Unfortunately, however, despite its theoretical attractiveness, so far such an approach has failed to yield improved results on the Norwegian data. This could possibly be due to errors in the syntactic analysis made by the Oslo-Bergen tagger, however, meaning that a possible direction for future research could be to conduct further experiments using manually corrected syntactic analysis.

8.9 The Centering-based algorithm

8.9.1 Overview of the algorithm

My implementation of a Centering-based algorithm is inspired by Brennan et al. (1987) (cf. section 7.2.4). However, their algorithm, as outlined in the paper, does not deal with binding of reflexives and possessives, nor does it give any recipe for handling coreference between markables within the same sentence. Furthermore, as pointed out by Brennan et al. themselves, their implementation does not aim at efficiency and performs a number of unnecessary computations. My own implementation, which avoids these shortcomings, can be summed up as follows:

(1) Rank all markables in S_n by syntactic function. As a starting point, I use the following order, taken from Brennan et al. (1987):

- SUBJECT
- OBJECT(S)
- others

In addition, I give special treatment to reflexives and possessives, in that they are ranked at the top of the hierarchy. The reason for this is that such markables are coreferent with the SUBJECT of the clause in which they occur, and hence belong at the same hierarchical level as the SUBJECT, but since they (normally)

occur *after* the SUBJECT, they, rather than the SUBJECT, should actually be selected as the antecedent (keeping in mind that an anaphor is always linked to its *closest* antecedent). Hence, SUBJECTs, reflexives, and possessives are ranked at the top of the hierarchy in a right-to-left order.

(2) Do the same with the markables in S_{n-1}. Note that, unless S_n is the first sentence in the text, in this step we can just re-use the result of step (1) at the preceding time step.

(3) Link any reflexives and possessives in sentence S_n to their respective binders, if any such binders can be found.

(4) Link the highest ranked of the remaining pronouns in S_n with the highest-ranked markable in S_n that is compatible with it in gender, number, and animacy. Gender and number compatibility are assessed using the same criteria as in the filters for the machine learning method that are outlined in section 8.5.1. Compatibility with respect to animacy is determined using the animate noun list obtained by the offline extraction approach described in section 6.4.

Note that linking non-reflexive anaphors to non-possessive markables within the same sentence is likely to be incorrect when the two also occur within the same clause (cf. Principle B of Chomsky's Binding Theory). However, the link may be correct if the two are found in different clauses within the same sentence, and experiments show that leaving out this step significantly decreases the performance of the AR system. Requiring that a clause boundary be found between the anaphor and the antecedent candidate also damages performance significantly, probably because many clause boundaries are invisible and cannot be detected by the grammatical tagger. This happens, for instance, in cases where the end of a subordinate clause is not marked by a comma.

If no compatible markable exists in S_n, the system performs a similar search for a suitable markable in S_{n-1}. If, after this, a suitable antecedent has still not been found, the algorithm selects the highest-ranked markable which is compatible in number and gender (but not animacy), if one exists. The reason for this additional step is that the lists of animate nouns gathered by the automatic procedure described in section 6.4 are necessarily incomplete, meaning that we should not rely too heavily on them.

If the pronoun is finally linked to a markable, the markable is blocked from being linked to other pronouns in S_n.

(5) Continue down the rank of pronouns in S_n. Repeat the process given in (4) for each pronoun as long as available markables in S_n or S_{n-1} can be found.

Centering Theory focuses on the existence of a single backward-looking center (C_b) in S_n and specifies how this center should be realized and what it should corefer with in the preceding sentence (or utterance) in order to make the discourse maximally coherent. When constructing a system for pronoun resolution, however, we cannot restrict ourselves to linking up the C_b; we have to deal with all of the pronouns in S_n (as, indeed, do Brennan et al. (1987)).

The algorithm outlined above accomplishes this while still adhering to the principles of Centering Theory. According to Constraint 3 of Brennan et al. (1987), the C_b of S_n should be the highest-ranked element of S_{n-1} that is realized in S_n. In the algorithm outlined above, we rank the markables in S_{n-1} by grammatical function. The pronoun in S_n that is linked to the highest-ranked markable in S_{n-1} (i.e., the C_b of S_n) will be the highest-ranked pronoun that is compatible with this markable in gender, number, and animacy, thus ensuring that Constraint 3 is not violated.

Furthermore, rule 1 in Brennan et al. (1987) states that, if any element in S_n is realized as a pronoun, then so is the C_b of S_n. By restricting ourselves to pronominal anaphora, we automatically determine the C_b to be a pronoun if one exists, while in sentences without any pronouns we do not make any attempt at determining the C_b at all.

Finally, the algorithm implicitly implements the transition hierarchy of Brennan et al. (1987), i.e., CONTINUING > RETAINING > SMOOTH SHIFT > ROUGH SHIFT[6]. Both CONTINUING and RETAINING require that $C_b(S_n) = C_b(S_{n-1})$, but CONTINUING also requires that $C_b(S_n) = C_p(S_n)$, i.e., that the backward-looking center of S_n is also the preferred, or highest-ranked, center of the sentence. By selecting $C_b(S_n)$ to be the highest-ranked pronoun that is compatible with $C_p(S_{n-1})$ in gender, number, and animacy, we implicitly favour a situation where $C_b(S_n) = C_p(S_n)$ (i.e., CONTINUING) over one in which this is not the case (i.e., RETAINING).

Also, following a CONTINUING, the algorithm will prefer CONTINUING or RETAINING over a shift. After a RETAINING, however, a shift will be preferred (since $C_p(S_{n-1})$, which will be linked to $C_b(S_n)$, will not be the same as $C_b(S_{n-1})$). On the face of it, this seems to go against the preferred transition hierarchy. Brennan et al. (1987) point out, however, that a shift might in fact be the most coherent choice after a RETAINING, since a RETAINING promotes some other element than $C_b(S_{n-1})$ to be $C_p(S_{n-1})$ and hence to be the most likely antecedent for $C_b(S_n)$, implying that a shift takes place. If $C_p(S_{n-1})$ is compatible in gender, number, and animacy with the most highly ranked element of S_n, i.e., $C_p(S_n)$, the two will be linked on the next time step, making $C_b(S_n)$ equal to $C_p(S_n)$. Since such an equality is what distinguishes a SMOOTH SHIFT from a ROUGH SHIFT, Brennan et al.'s preference for SMOOTH SHIFT is honoured by the algorithm described here.

[6]In Brennan et al.'s terminology, SMOOTH SHIFT is called SHIFTING-1 and ROUGH SHIFT is called SHIFTING.

8.9.2 The definition of *utterance*

The concept of *utterance* plays an important role in the rules and principles of Centering Theory, since utterances are the units which contain the *centers* that are the crucial elements of CT, but this concept is nevertheless not explicitly defined in the original formulations of the theory. Naturally, an actual implementation of an AR system based on CT needs to be explicit with respect to what counts as an utterance, or an *update unit*, which it is sometimes called in this connection, because of the emphasis on its role as the unit in which centers are updated.

Several proposals for suitable definitions of update units have appeared in the literature. The proposal made by Kameyama (1993, 1998) is summed up by Miltsakaki (1999, p. 129) as follows: "conjoined and adjoined tensed clauses form independent units whereas tenseless subordinate clauses, report complements and relatives [sic] clauses belong to the update unit containing the matrix (superordinate) clause." Eugenio (1990, 1998) offers what is claimed to be empirical evidence for Kameyama's view that tensed adjunct clauses should be treated as update units. Miltsakaki (1999), however, discusses these earlier analyses and argues that the evidence (including her own Greek and Japanese data) rather points to the traditional sentence as the most appropriate update unit. Following Miltsakaki, I treat sentences as update units in my own implementation for Norwegian, but with the modification mentioned above, viz. that antecedents can be found within the same sentence as the anaphor.

8.9.3 Results and error analysis

When evaluated on the default test set of anaphors, i.e., *han*, *hun*, and *den*, the Centering Theory-based approach results in a very modest accuracy of 38.72% on the development corpus, which is far below the performance achieved by the machine learning approach and, as we will see later in this chapter, also far behind the factor/indicator-based approach (both differences are significant at the 0.1% level). If *de* is added to the evaluation, performance decreases slightly to 37.46%. If, on the other hand, the neuter singular pronoun *det* is added to the evaluation set, the accuracy reaches 52.63% with *de* included and 55.71% without *de*.

On the test corpus, the performance of the Centering Theory approach is actually somewhat better than on the development corpus but still significantly worse than both the machine learning approaches and the factor/indicator-based approach, with the differences being significant at the 0.1% level. The performance of the Centering Theory-based approach on various pronouns and pronoun combinations in the development and test corpora is summarized in Table 8.38.

	Development corpus	Test corpus
han/hun	39.41	41.42
den	32.08	55.56
de	32.06	44.44
han/hun/den	38.72	42.07
han/hun/den/de	37.46	42.33
han/hun/den/de/det	52.63	54.41
han/hun/den/det	55.71	55.34

Table 8.38: Results for the Centering Theory-based approach on the development and test corpora.

These results agree with the finding by Tetreault (2001) that the algorithm described by Brennan et al. (1987) performs much worse than alternative approaches on English newspaper and fiction material (cf. section 7.2.4).

The reasons for the poor performance of the CT-based approach on the fiction material probably include the following:

- Lack of relevant information. A pronoun resolution approach that is based on Centering Theory relies on three factors. The first is the ordering of the centers (i.e., markables) in a sentence or utterance (normally by syntactic function, as I have done here, but information status can also be used; cf. Strube (1998); Strube and Hahn (1996, 1999)). The second factor is the ranking of transition types. Finally, anaphor–antecedent relations have to be restricted to those cases in which the two are compatible in number, gender, and animacy.

  In contrast, the other types of knowledge-poor approaches that are commonly used, i.e, factor/indicator-based approaches and machine learning approaches, are both able to utilize a much wider range of information about the anaphor, the antecedent candidate, the relationship between the two, and the context in which they occur. Hence, it is hardly surprising that both of these approaches turn out to perform much better on the Norwegian data, as we saw earlier in this chapter.

- Centering Theory, at least in its original formulation, specifies that a backward-looking center in utterance U_n must be realized in the immediately preceding utterance U_{n-1} (cf. the *Locality of* $C_b(U_n)$ claim of Grosz et al. (1995, p. 11)). In the Norwegian material, however, this is often not the case. The antecedent of a pronoun is often found many sentences earlier, and in particular there are many coreference chains consisting of identical pronouns which are separated by a large number of sentences.

  Since this is a characteristic of the training material as well as the test material, the machine-learning approach often selects the closest identical pronoun as

antecedent, regardless of the number of sentences separating the pronouns and the number of other eligible antecedent candidates in-between. The CT-based approach, on the other hand, does not use lemma matching for resolution of pronominal anaphora, and thus it is likely to select a closer candidate which seems eligible when disregarding semantic or pragmatic factors or world knowledge. Characteristics of the distribution of anaphor–antecedent distances in the Norwegian material are further discussed in section 8.11.

- Insufficient information about animacy and natural gender. Despite the animacy information that was gathered using the methods described in chapter 6, as well as the information provided by the earlier work referred to in that chapter, it is still the case that animacy information is lacking for a large proportion of nouns that are encountered in running text. Fiction texts tend to contain a high proportion of distinctly animate anaphors. On the other hand, most nouns denote inanimate entities. Thus, animacy information is required to pick out those nouns that do in fact denote animate entities and which are therefore eligible as antecedents for these anaphors. The current shortage of such information is therefore likely to contribute to the relatively low performance level of an algorithm such as the CT-based approach, which relies more heavily on animacy information than alternative approaches due to its lack of other information sources.

  As for natural gender, which is required to decide whether a given markable is a suitable antecedent for either *han* or *hun*, this is not available for common nouns, and for proper nouns it is only available for relatively frequent first names (i.e., those first names that are listed in the tagger lexicon).

8.10 The factor-based approach

In comparison with Centering Theory-based approaches, systems that are based on *factors* or *indicators*, like the ones presented by Lappin and Leass (1994), Kennedy and Boguraev (1996), and Mitkov (1998, 2002), are not so much the outcome of holistic linguistic theories. Rather, they bring together a set of seemingly useful aspects of anaphors and potential antecedents and weight them in the way that yields the best overall results, normally as the result of trial and error. Hence, these systems may seem less clearly linguistically motivated, but on the other hand they are able to make use of a more diverse range of information types, and therefore often lead to better results.

The factor/indicator-based approach employed in the current work is a slightly modified reimplementation of *ARN*, the AR system for Norwegian described by Holen (2006). Gordana Ilić Holen's system was inspired by earlier factor-based approaches,

in particular MARS (Mitkov, 2002) and RAP (Lappin and Leass, 1994). However, Holen's system makes a number of adaptations to the factor weightings found in those earlier works, mainly due to differences between the information structure of Norwegian and English sentences. My reimplementation is based on the description of ARN given in Holen (2006) as well as personal communication with the author. The lists of animate and inanimate nouns used by ARN were graciously provided by Holen.

8.10.1 Factors

The factors used in Holen's system were listed in section 7.2.7, and are repeated here for convenience:

- **Number/gender/animacy**: Match or mismatch between anaphor and antecedent candidate on number, gender, and animacy is awarded points according to Figure 8.2 on page 249.
- **Reference proximity**: Proximity between anaphor and antecedent is awarded in the following way:
  - Antecedent found in the same sentence: 100 points
  - Antecedent found in the previous sentence: 50 points
  - Antecedent found in earlier sentences: 0 points
- **Boost pronoun**: Pronominal candidates are awarded 75 points.
- **Direct object preference**: Direct object candidates are awarded 50 points.
- **Adverbial phrase penalization**: Candidates in adverbial phrases are penalized with -50 points.
- **Syntactic parallelism**: Candidates with the same syntactic function as the anaphor are awarded 50 points.
- **Section heading preference**: Candidates that occur in the heading of the section in which the anaphor occurs are awarded 50 points.
- **Indefiniteness penalization**: Indefinite candidates are penalized with -25 points.

8.10.2 Modifications to ARN

My reimplementation of ARN includes a number of modifications to the original system:

- Reflexives and possessive pronouns are included as potential antecedents. These were not originally handled by ARN, because reflexives and possessives were not included in the coreference chains in the version of the BREDT data that was used by Holen.

  Instances of *seg* "himself/herself/itself/themselves" that are erroneously analysed as past tense of the verb *sige* "sink slowly" by the grammatical tagger are included as potential antecedent candidates despite being tagged as verbs. Due to the high frequency of the pronoun *seg* and the low *actual* frequency of the homographous verb form in Norwegian, I make the simplifying assumption that all occurrences of *seg* are in fact pronouns. Evaluation of the system confirms that this leads to improved performance.

- Links between incompatible pronouns are filtered out. In effect, this means that singular pronouns can only be linked to other pronouns of the same kind; no linking of, e.g., *han* "he" to *hun* "she" or *det* "it (neut.)" to *den* "it (masc./fem.)", etc. is permitted. The plural pronoun *de* "they" is allowed to be linked to any pronoun, because it may have either a plural antecedent, which may be another occurrence of "de", or a group of singular (or plural) antecedents, each of which may take the form of any pronoun.

- The named entity recognizer described in chapter 4 is used to determine whether proper names denote human beings. This information is incorporated into the *Number/gender/animacy* factor.

- For the additional benefit of the *Number/gender/animacy* factor, animacy information gathered from Web queries as described in chapter 6 is utilized as a complement to the manually crafted lists of animate and inanimate nouns used by the original system.

8.10.3 Results

When tested on the pronouns *han*, *hun*, and *den* (as in Holen (2006)'s work), the factor-based approach reaches an accuracy of 68.92% on the development corpus, which is much better than the Centering Theory-based approach but at the same time considerably lower than the 74.60% obtained by TiMBL (both differences are significant at the 1% level). It is also somewhat lower than the 70.2% accuracy reported by Holen on the fiction part of her test corpus (which is drawn from the same material as the development corpus used in the present work). One reason for this may be that, although the fiction material submitted to the two systems does consist of the same texts, it was not split into training and test (or development) corpora in the same way (the split for my reimplementation was the same as the one used for the machine

learning and centering-based algorithms), and hence we cannot expect exactly the same performance.

More importantly, however, although Holen also got her material from the BREDT project, she used an earlier version of the data which was annotated before the final guidelines for the project were finalized. The version that I am using has been re-annotated according to the final guidelines. There are a number of differences between the two versions, the most important one for the present purposes being that reflexives and possessive pronouns that were not included in the coreference chains in the earlier version are now taken to be part of the chains (cf. section 8.10.2).

Hypothesizing that this could be a likely cause for the difference in performance, I had a look at ARN's treatment of two of Holen's test files (files 11 and 12; cf. Holen 2006, p. 81) and compared it to the output from my own system on the same data[7]. Out of a total of 47 errors made by my system on these files, I focused on those 12 cases where ARN had made the correct choice.

In three of these cases, there was an error in the annotated data—in other words, the system actually made the correct decision but the gold standard was mistaken. Of the remaining cases, eight involved reflexives or possessive pronouns which were either missed or mistakenly selected, while in the last case the correct antecedent was given as an apposition to a SUBJECT whereas in the earlier version of the data the SUBJECT itself was taken to be the antecedent (this SUBJECT was the markable selected by both ARN and my own system).

Hence, all of the mistakes made by my system but not by ARN on these two files can be traced back to changes in the annotation used in the two versions. In all of these cases, therefore, the original ARN would have made the same mistakes if it had been fed the same version of the data, and this indicates that the differences in annotation constitute an important cause of the difference in performance levels between the original and the reimplemented version of ARN.

When tested on the test corpus, which is not part of the BREDT material, the ARN reimplementation reaches accuracy scores of 66.12% on *han/hun/den*, 62.71% on *han/hun/den/de*, 70.01% on *han/hun/den/det/de*, and 73.22% on *han/hun/den/det*. As was the case with the development corpus, the TiMBL-based approach with feature percolation and backup antecedents performs better than the ARN reimplementation on the test corpus, with all differences being significant at the 1% level ($p \ll 0.01$). The performance of the ARN reimplementation is summarized in Table 8.39.

[7] The output from ARN was graciously provided by Holen.

	Development corpus	Test corpus
han/hun	71.57	66.23
den	43.40	63.89
de	38.93	35.35
han/hun/den	68.92	66.12
han/hun/den/de	63.26	62.71
han/hun/den/de/det	70.41	70.01
han/hun/den/det	75.11	73.22

Table 8.39: Results for the factor/indicator-based approach on the development and test corpora.

8.11 Some genre-specific challenges

Fiction material presents some challenges for AR which tend to be less pronounced in most other genres. The experiments reported here highlight the following more of less genre-specific problems:

- *People are often introduced by pronouns.* Very often, the first mention of the participants in a story has the form of a pronoun such as "he", "she", or "they". Thus, in fiction material, these pronouns may be used non-anaphorically, something which is very uncommon in other types of (written) text, and which may lead the system to propose antecedents where no antecedents should be found. For example, the development corpus used in the present experiments starts in the following way:

  (10) Tilfeldig møte på Heathrow. Hovedveien ut mot flyplassen var støvet. Husene$_{12}$ langs ruten lå tilbaketrukket mellom tørre, brunsvidde hageflekker$_{21}$, som ingen kunne gjøre noe med fordi det var vanningsforbud$_{32}$. De$_{34}$ kjørte bak en fullastet buss med et reiseselskap, trolig på vei til tusenkroners pakkeeventyr på et fjernt, men sikkert like brunsvidd sted ved Middelhavet.
  "Accidental meeting at Heathrow. The main road to the airport was dusty. The houses$_{12}$ along the way were receded between dry, scorched [garden patches]$_{21}$, which no-one could do anything about because there was a [watering ban]$_{32}$. They$_{34}$ were driving behind a crowded bus with a travelling company, probably on their way to some thousand *kroner*'s charter adventure in a remote, but probably equally scorched, place by the Mediterranean Sea."

  The pronoun *de*$_{34}$ is used to introduce some participants of the story and therefore does not have an antecedent. However, an AR system will typically expect

an antecedent for this kind of pronoun, and since there are several markables in the preceding sentence that agree with the pronoun in number[8], the system will probably (wrongly) select one of these as the antecedent of de_{34}.

- *"They" may be used in a non-specific way.* In fiction, as in spoken language (Strube and Müller, 2003), the pronoun "they" may refer to some group of people which is only vaguely specified, like for instance "the government" ("They have banned smoking in restaurants") or "people in general" ("They say a famous person was once living in this house."). In such cases, the pronoun does not have an antecedent, but the AR system will nevertheless select one if it finds a suitable candidate. Consider the following example from the development corpus:

  (11) Folk$_{216}$ svettet, bannet og ble aggressive, og som toppen på kransekaka hadde de$_{230}$ begynt å rasjonere på vannet.
  "People$_{216}$ were sweating, cursing and becoming aggressive, and to top it off they$_{230}$ had started to ration the water."

  To an AR system, *folk*$_{216}$ will look like a perfect antecedent candidate for de_{230}. In order to discover that it is in fact not a suitable candidate, the system would need to understand that the people rationing the water cannot be the same as those becoming aggressive because of the water rationing—a kind of understanding that would be very hard to build into an AR system.

  What the system also would really need to understand, however, in order to avoid looking for other antecedent candidates, is that the pronoun de_{230} refers to a group of people that are external to the story described by the text, and therefore should not be assigned any antecedent at all. Clearly, this kind of understanding would be equally hard to build into the system, if not harder.

- *Animate pronouns do not always corefer with the closest suitable markable.* In other types of text, if we have an animate pronoun such as "he" or "she" and a single suitable antecedent candidate in the immediately preceding context, we can be fairly certain this candidate is in fact the antecedent of the pronoun. In fiction texts, on the other hand, the pronoun may refer to the main character of the story rather than coreferring with the said markable. The following example from the development corpus illustrates the kind of situation I am referring to:

  (12) –Ut og reise, sir? Den magre cockneysjåføren$_{255}$ var åpenbart nysgjerrig

[8] *vanningsforbud* "watering ban" can be either singular or plural, and since world knowledge is needed in order to decide that it is in fact singular in this context, the tagger—which does not possess such knowledge—has left the word ambiguous between singular and plural.

av natur. –Ja. Han$_{266}$ orket ingen konversasjon i den kvelende atmosfæren.
"–Going on a trip, sir? The skinny [cockney driver]$_{255}$ was clearly curious by nature. –Yes. He$_{266}$ could not stand a conversation in the suffocating atmosphere."

Pragmatic knowledge tells us that the person who cannot stand a conversation must be different from the curious and talkative cockney driver. Thus, we can infer that *han*$_{266}$ must rather refer to the main character of the story, who was in fact last mentioned five sentences before this excerpt. To an AR system without pragmatic knowledge, however, *cockneysjåføren*$_{255}$ looks like a very good antecedent candidate for *han*$_{266}$.

The point I will argue here is that, in other genres, we would be much less likely to find cases like this where a "he" or a "she" does not corefer with such a seemingly perfect markable but rather refers to a kind of "omnipresent" character that may not have been mentioned in a long time.

In other cases, the referent of the pronoun has to be inferred from the context, for example as being the person behind a recent utterance:

(13) Thomsen$_{300}$ karret seg ut og ventet til sjåføren$_{307}$ hadde plasert [sic] den ene kofferten hans$_{313}$ på fortauet. –Fem pund femti, sir. Han$_{324}$ fikk en tier og lyste opp.
"Thomsen$_{300}$ got out and waited until the driver$_{307}$ had placed one of his$_{313}$ suitcases on the pavement. –Five pounds fifty, sir. He$_{324}$ got a tenner and lightened up."

In this sequence, both *Thomsen*$_{300}$, *sjåføren*$_{307}$, and *hans*$_{313}$ are compatible with *han*$_{324}$ in gender, number, and animacy. Furthermore, both *Thomsen*$_{300}$ and *sjåføren*$_{307}$ are SUBJECTs like *han*$_{324}$, hence matching on both the *Syntactic function of antecedent* and the *Syntactic parallelism* features. In other words, both *Thomsen*$_{300}$ and *sjåføren*$_{307}$ look like good candidates for being the antecedent of *han*$_{324}$ (while *hans*$_{313}$ can be linked to *Thomsen*$_{300}$ through Binding Principle B (Chomsky, 1981, 1986), effectively incorporating it into the same coreference chain as *Thomsen*$_{300}$).

In order to decide on the correct antecedent, a human reader needs to perform a pragmatic interpretation of the sequence. Since the driver is the one who is most likely to ask for money as well as the one most likely to use the word *sir* when addressing the other person, the reader will probably conclude that the driver is the one making the utterance requesting money.

Consequently, in the next sentence, when some "he" receives more money than was asked for and lightens up because of this, the reader is likely to conclude that "he" refers to the driver that was making the request, and hence correctly link han_{324} to $sjåføren_{307}$. An automatic AR system, on the other hand, does not possess the pragmatic knowledge needed to make these kinds of inferences and hence will have difficulties choosing the correct antecedent.

- *Many antecedents are located far away from the anaphor.* Compared to the newspaper texts used in the MUC-6 competition, a large proportion of the antecedents in the BREDT corpus are located relatively far away from the anaphor. This makes it harder to find the antecedents without accidentally selecting a wrong candidate between the anaphor and the actual antecedent. However, it is not clear whether this is due to different genres or to the language difference. Consider Table 8.40, which displays some information about the distances between anaphors and antecedents in the Norwegian BREDT corpus, and Table 8.41, which summarizes the information about anaphor–antecedent distances for pronominal anaphors given by Hoste (2005) for some of her English MUC-6 data and her Dutch KNACK-2002 data.

Location of antecedent	BREDT
In same sentence as anaphor	37.6%
In previous sentence	41.4%
Two sentences before	11.16%
More than two sentences before	9.81%

Table 8.40: Summary of anaphor–antecedent distances in the Norwegian BREDT data.

Location of antecedent	MUC-6	KNACK-2002
In same sentence as anaphor	73.0%	41.1%
In previous sentence	22.7%	29.2%
Two sentences before	2.2%	6.9%
More than two sentences before	2.1%	22.8%

Table 8.41: Summary of anaphor–antecedent distances in the English and Dutch data used by Hoste (2005).

The MUC-6 corpus consists of newspaper articles from the *Wall Street Journal*, while the KNACK-2002 corpus contains articles from a Flemish weekly magazine about national and international current affairs. Hence, both of these corpora belong to the news text genre, but they display very different distributions of antecedents. The distribution in the BREDT data is more similar to KNACK-2002 than to MUC-6. The fact that both of Hoste's corpora consist of news texts could point to a difference between English and Dutch with regard to antecedent

distance, with Norwegian being more similar to Dutch than to English in this respect. An alternative explanation might be that *KNACK*, despite being a news magazine, is nevertheless written in a style more similar to fiction than to the *Wall Street Journal*. Further studies are needed in order to shed light on this question.

Regardless of the explanation for the difference in antecedent distribution, it does create additional challenges for the Norwegian system compared to the MUC systems. A preference for close antecedents is reflected in many aspects of existing AR systems. For example, Hobbs (1978) selects the closest available antecedent subject to syntactic restrictions, and many other systems include a distance feature that favours close antecedent candidates. Since proximity has proved to be an important factor in earlier AR work, these systems are in effect able to restrict the search for antecedents to a relatively short text span, which hugely simplifies the task. On the other hand, the very different distribution of anaphor–antecedent distances in the Norwegian fiction data means that the system presented here needs to consider a much larger number of potential antecedents, and it is therefore faced with a considerably harder decision task.

The importance of this point becomes particularly clear from data given by Hoste (2005, p. 149) on the performance of her classifiers on different anaphor–antecedent distances in the MUC-6 data. In Hoste's experiments, both TiMBL and RIPPER show a dramatic performance drop with distance for cases where the anaphor is a pronoun. For a distance of two sentences, TiMBL performs around $F_{\beta=1}$ = 25-30 and RIPPER has an $F_{\beta=1}$ of about 30-35. For a distance of three, the score for TiMBL is around 20-25, while RIPPER scores around 15-20. In cases where the antecedent is even further removed from the anaphor, the TiMBL scores are about 15-20, while those for RIPPER are around 10-15. Partly for this reason, Hoste's test set only includes antecedents located maximally two sentences before a pronominal anaphor (Hoste, 2005, p. 147). Thus, with approximately 20% of the antecedents in the BREDT corpus being at least two sentences away from the anaphor (and with all antecedents included in the test set, regardless of distance), the challenge presented by the BREDT data is considerably harder than that presented by the MUC-6 data.

8.12 Future work

The performance of the best TiMBL-based system presented in this chapter compares favourably with earlier work on Norwegian anaphora resolution. Nevertheless, I envision a number of strategies for future work which might lead to further improvements to the system. This section describes a number of such areas of improvement.

8.12.1 General coreference resolution

Although the present work focuses on pronominal anaphora, the AR system is constructed in such a way that it can easily accommodate general coreference resolution. Results of preliminary experiments on this task have not been impressive, however. This is hardly surprising, for several reasons. First, the performance figures for full-NP anaphora that have been reported in the literature are far lower than those for pronominal anaphora. Furthermore, a number of the features introduced by Soon et al. (2001) that are potentially useful for coreference resolution have been left out of the present system. An interesting direction for future work would be to make the system better equipped for general coreference.

Using co-occurrence statistics

A kind of information that would seem useful for general coreference resolution is co-occurrence statistics involving anaphors and potential antecedents, either directly between the two or indirectly in the form of co-occurrence as arguments of the same verb or complements of the same preposition.

In fact, a number of such features have already been implemented for the present system, using a database of co-occurrence information extracted from the Oslo Corpus by Chris Biemann. Presently, however, such features actually damage the performance of the system, which is hardly surprising given that we focus on pronominal anaphora; co-occurrence statistics between pronouns and lexical words are more likely to introduce random noise into the system than to contribute any kind of useful information. For coreference resolution of full-NP anaphora, however, this kind of information can be expected to be highly valuable.

8.12.2 Separating referential and pleonastic *det*

Like English *it*, Norwegian *det* constitutes a problem because of the fact that it can be used either referentially or pleonastically. If it is referential, *det* should have an antecedent in the text, unless it is used only deictically to refer to some entity in the non-linguistic context. If, on the other hand, it is pleonastic, it does not have a referent, and hence an AR system should not attempt to find an antecedent for it. In these cases, *det* fills a required slot in a syntactic construction, but is void of semantic content.

Some examples of pleonastic *det* are given in (14).

(14) a. Det regner.
it rains
"It is raining."

b. Det sitter en mann på trappa.
it sits a man on stairs-the
"A man is sitting on the stairs."

c. Det ble sunget og danset hele natta.
it was sung and danced all night
"There was singing and dancing all night."

d. Det er John som har gjort det.
it is John who has done it
"John (is the one who) did it."

In (14-a), *det* fills the SUBJECT slot which is required in all declarative main clauses in Norwegian, although the verb has zero valency and hence does not require (or permit) any participant constituents. Example (14-b) shows an example of a common Norwegian construction in which the Agent participant is demoted to OBJECT and the SUBJECT position is filled by a semantically void *det*; Sveen (1996) gives a thorough treatment of this kind of construction as well as the impersonal passive construction exemplified in (14-c). Finally, (14-d) illustrates a cleft, a kind of construction which is much more common in Norwegian than in English (Gundel, 2002).

Since Norwegian has a larger range of constructions that involve pleonastic *det* than is found in English (Holen, 2006), and since these constructions are so frequent (particularly the constructions in (14-b) and (14-d)), the problem of separating referential from pleonastic uses emerges more often in Norwegian than in English. On the other hand, Norwegian AR may take advantage of the Norwegian gender system to make this task more tractable. *Det* is a neuter pronoun, and in those cases where it is referential and does corefer with an NP (rather than, say, a clause, a VP, or an infinitival construction), the coreferent NP must also be neuter. This fact can be utilized both in the process of determining referentiality and in the task of finding an antecedent once a pronoun has been determined to be referential.

Many researchers working on English AR systems have elected to ignore pleonastic uses of *it* and have simply removed them from their test corpora. Lappin and Leass (1994), on the other hand, do not ignore these cases, but argue instead that they should be handled by a different mechanism in a preprocessing step prior to the actual AR procedure. I believe that this is a sensible position, since the process of separating referential from pleonastic uses of pronouns may rely on different factors than those that are involved in the process of determining an antecedent for a pronoun known to be referential.

Thus, the factors used in a Lappin and Leass-type AR system or the features used in a machine learning AR system may not be suitable for determining whether or not a certain pronominal token is referential. Lappin and Leass (1994) and Denber (1998) instead postulate sets of grammatical constructions that are used to identify pleonastic uses of *it*, while Evans (2001) applies a machine learning approach to the task, training

a memory-based learner on features that encode a wide variety of information about the words in the vicinity of *it*. Evans' system has also been incorporated into MARS (Mitkov, 2002). Boyd et al. (2005) combine the grammatical approach and the machine learning approach, using grammatical constructions, part-of-speech information, and lexical information as features for a memory-based learner. They achieve an accuracy of 88%, with a precision of 82 and a recall of 71. Similar techniques might be used in future versions of the Norwegian system in order to tackle the problem of separating referential and pleonastic occurrences of *det*.

8.12.3 Identifying constituents of subordinate clauses

Previous research has indicated that the arguments of the main clause are more accessible as antecedent candidates than arguments of subordinate clauses (Miltsakaki, 1999). Hence, being able to distinguish between arguments of the main clause and arguments of subordinate clauses is an important requirement for making the best possible antecedent choices. Unfortunately, the Oslo-Bergen tagger does not perform a hierarchical syntactic analysis. If a sentence contains multiple clauses, the tagger will give information on which SUBJECTs, OBJECTs, etc. occur in the sentence, but it will not tell us which of these are arguments of the main clause and which are arguments of subordinate clauses.

In order to alleviate this problem, one could implement a heuristic method for identifying the constituents of some subordinate clauses. This could be done by taking advantage of the fact that Norwegian is a V2 language, meaning that the finite verb of a declarative main clause is always the second constituent counting from the start of the clause. This implies that, if the first constituent of a declarative sentence is a subordinate clause (which can be identified by an initial subjunction), this clause must be the one and only argument of the main clause verb. Since the subordinate clause itself must contain a verb, we know that the first verb found cannot be the verb of the main clause; only the next verb to the right may possibly be the main clause verb (although that may also belong to a subordinate clause). Hence, if a sentence starts with a subjunction, all arguments that are found before the second verb in the sentence must belong to some subordinate clause rather than to the main clause, and they may therefore be dispreferred as antecedent candidates for anaphors occurring later in the text.

8.13 Conclusions

In this chapter, I have presented an anaphora resolution system for Norwegian fiction material, mainly by using machine learning approaches.

The system presented in this chapter is substantially improved with respect to the earlier versions described in Nøklestad et al. (2004), Nøklestad and Johansson (2005), and Nøklestad et al. (2006). The current system uses a different set of machine learning features; the implementation of the filters (cf. section 8.5.1) has been tweaked; and a number of support modules have been developed (i.e., the named entity recognizer, the PP attachment disambiguator, and the animacy detection procedures presented in earlier chapters).

I have compared three different machine learning algorithms (memory-based learning, maximum entropy modelling, and support vector machines) as well as an approach based on Centering Theory and one using indicators or factors. The approach that achieves the best performance uses memory-based learning with default parameter settings, feature percolation, and backup antecedents. This system reaches an accuracy of 74.60% on *han/hun/den* on the development corpus, which is considerably better than the accuracy achieved by the only earlier anaphora resolution system for Norwegian (Holen, 2006) on material drawn from the same corpus. Using 10-fold cross-validation, the TiMBL-based system achieves an accuracy of 70.58% on these pronouns, while its accuracy on a separate test corpus is 72.42%.

Thus, both the cross-validation accuracy and the test corpus accuracy of the TiMBL-based system are considerably lower than the accuracy obtained on the development corpus, and this seems to indicate that its performance on the development corpus has benefited from the fact that the system has been fine-tuned on this material. Nevertheless, even the cross-validation accuracy is better than the results obtained by the Centering Theory and factor/indicator-based approaches on the development corpus (38.72% and 68.92%, respectively).

In fact, if we simulate cross-validation with these alternative approaches by running them on the entire corpus, the accuracy scores obtained are significantly lower than the cross-validation accuracy of the memory-based system: for the Centering Theory approach, the difference is significant at the 0.1% level (accuracy: 41.25%; $p \ll 0.001$), while for the factor/indicator approach it is significant at the 5% level and bordering on significance at the 1% level (accuracy: 67.32%; $p \leq 0.0111$). Furthermore, on the test corpus, which does not contain any material from the corpus that was used to develop both Holen's system and the machine learning approaches described here, the TiMBL-based approach outperforms both the factor/indicator approach and the Centering Theory approach with differences that are significant at the 1% level.

I have implemented three different classifiers for different kinds of pronouns, and this turns out to give significantly improved results, at least for the *den* pronoun. I have also introduced features that approximate principles A and B of Chomsky's Binding Theory. The first of these features receives a very large gain ratio weight, although it does not by itself yield significant performance improvements.

Finally, and importantly, I have developed three support modules that in turn provide information to the anaphora resolution system: a named entity recognizer, an animacy detector, and a PP attachment disambiguator. The first two turn out to be beneficial for my AR system (cf. section 8.7), while the latter fails to have a significant impact on the performance of the system in spite of obtaining high gain ratio machine learning weights.

```
David       <ord ne="pe" suff="vid" id="5228" genus="m" lemma="David" syn="np"
                 pos="prp"><ant type="r" id="5187-5188"/>
                 David</ord>
,           <ord suff="," id="5229" lemma="$," syn="sgr" pos="cma">
                 ,</ord>
said        <ord suff="sa" id="5230" lemma="si" syn="fv" pos="v">
                 sa</ord>
she         <ord suff="hun" id="5231" num="sg" genus="f" lemma="hun" syn="subj"
                 pos="pron"><ant type="r" id="5213"/>
                 hun</ord>
softly      <ord suff="avt" id="5232" num="sg" genus="n" lemma="lav" syn="adv"
                 pos="a">
                 lavt</ord>
,           <ord suff="," id="5233" lemma="$," syn="unk" pos="cma">
                 ,</ord>
to          <ord suff="til" id="5234" lemma="til" syn="adv" pos="p">
                 til</ord>
himself/herself/itself/themselves
            <ord suff="seg" id="5235" lemma="seg" syn="pfill" pos="pron">
                 <ant type="r" id="5231"/>
                 seg</ord>
self        <ord suff="elv" id="5236" lemma="selv" syn="det" pos="d">
                 selv</ord>
,           <ord suff="," id="5237" lemma="$," syn="unk" pos="cma">
                 ,</ord>
almost      <ord suff="ten" id="5238" lemma="nesten" syn="adv"
                 pos="adv">
                 nesten</ord>
like        <ord suff="som" id="5239" lemma="som" syn="adv" pos="p">
                 som</ord>
a           <ord suff="et" id="5240" num="sg" genus="n" lemma="en" syn="det"
                 pos="d">
                 et</ord>
question    <ord suff="mål" id="5241" num="sg" genus="n" lemma="spørsmål"
                 syn="pfill" pos="ni">
                 spørsmål</ord>
.           <ord suff="." id="5242" lemma="$." syn="unk" pos="dot">
                 .</ord>
-           <ord suff="xsep" id="5243" lemma="$-" syn="unk" pos="sep">
                 -</ord>
he          <ord suff="han" id="5244" num="sg" genus="m" lemma="han" syn="subj"
                 pos="pron"><ant type="r" id="5228"/>
                 Han</ord>
is          <ord suff="er" id="5245" lemma="være" syn="fv" pos="v">
                 er</ord>
on          <ord suff="på" id="5246" lemma="på" syn="adv" pos="p">
                 på</ord>
the-kitchen <ord suff="net" id="5247" num="sg" genus="n" lemma="kjøkken"
                 syn="pfill" pos="nd"><ant type="r" id="5022"/>
                 kjøkkenet</ord>
,           <ord suff="," id="5248" lemma="$," syn="sgr" pos="cma">
                 ,</ord>
said        <ord suff="sa" id="5249" lemma="si" syn="fv" pos="v">
                 sa</ord>
Sonja       <ord ne="pe" suff="nja" id="5250" genus="f" lemma="Sonja" syn="subj"
                 pos="prp"><ant type="r" id="5205"/>
                 Sonja</ord>
```

Figure 8.1: An example of the XML format used in the anaphora resolution task, as discussed in section 3.2.

Pronouns			***den***	***han***	***hun***
Nouns	Class 1 *not human*	m	100	-100	-100
		f	100	-100	-100
		n	0	-100	-100
	Class 2 *human of indeterminable gender (male?)*		-50	100	75
	Class 3 *human male*		-50	100	0
	Class 4 *human female*		-50	0	100

Figure 8.2: Points awarded based on gender and animacy in ARN, as discussed in section 8.10.1. Adapted from Holen (2006) (points for pronouns that are not evaluated (*De* and *de*) have been removed).

Chapter 9

Conclusions

In this thesis, I have described an automatic anaphora resolution system for Norwegian which is able to accommodate a range of different approaches to the AR task. My main focus has been on the use of various machine learning methods, with particular weight on memory-based learning, and my system is the first AR system for Norwegian to employ machine learning methods.

However, I have also presented alternative knowledge-poor approaches that are based on Centering Theory or a set of factors or indicators. The modular nature of the AR system has made it easy to compare the different approaches on the same data, guaranteeing the same morphological and syntactic preprocessing and the same set of markables for the algorithms to choose from. The experiments presented in this thesis constitute, to my knowledge, the first systematic comparison of all these knowledge-poor approaches on identical data—earlier work such as Ng and Cardie (2002b, 2003b), Ng (2005), Hoste (2005), and Hendrickx et al. (2008) has compared various machine learning methods but has not included approaches that are not based on machine learning.

In addition to the anaphora resolution system, I have developed three support modules. The tools and resources developed for this thesis are the following:

- An anaphora resolution system based on
  - machine learning methods:
    - memory-based learning
    - maximum entropy modelling
    - support vector machines
  - Centering Theory
  - a factor/indicator-based approach

- A named entity recognizer
- A PP attachment disambiguator
- Methods for detecting animacy in nouns

Performance on anaphora resolution

Experiments on fiction material show that the AR system developed here performs favourably in comparison with the only previously existing automatic AR system for Norwegian, i.e., the ARN system presented by Holen (2006). Holen's results are impressive and show that it is possible to achieve a high accuracy on Norwegian anaphora resolution without the aid of machine learning techniques to determine suitable weights for the different information sources used by the system. Nevertheless, the accuracy of 74.60% obtained by the machine learner on the development corpus is significantly better than the 68.92% accuracy achieved by my ARN reimplementation, and it is also much better than the 70.2% obtained by Holen's original system on the older version of data drawn from the same corpus. The improvement over Holen's original system cannot be tested for statistical significance because of the differences in test material, but a 4.4 percentage point improvement appears solid.

Furthermore, the machine learning system significantly outperforms the ARN reimplementation on a separate test corpus that does not contain any material from the BREDT corpus that was used to develop both the machine learning approaches and Holen's original system. Thus, it seems that a machine learning approach holds definite advantages over one based on manually determined weights on this task.

Unlike earlier memory-based systems for anaphora resolution, my system, when used with the memory-based learning algorithm, employs different classifiers for different pronominal anaphors. The classifiers develop relatively large differences in gain ratio weights for the various machine learning features, illustrating a large variability between the features with respect to their importance for the different anaphors. By letting the different pronouns be handled by separate classifiers, I am able to improve the performance of the system significantly (albeit only on the resolution of *den*).

In my AR experiments, memory-based learning with default parameter settings and MaxEnt modelling with automatically optimized parameter settings perform best on the development corpus, while TiMBL or MaxEnt with default settings and SVM with optimized settings perform equally well on the test corpus.

The good performance of TiMBL and, to a lesser extent, MaxEnt, is a general trend found in most of the machine learning experiments described in this thesis: in almost all cases, memory-based learning works equally well as or better than MaxEnt modelling, while SVM, in those cases where it is applied, i.e., PP attachment and anaphora resolution, tends to perform considerably worse than the other two. The

only exception occurs when optimized parameter settings are used on the test corpus for AR. In this case, SVM outperforms TiMBL, but only if TiMBL is also forced to use optimized rather than default parameters; if TiMBL is allowed to run with default parameters, the difference between the two is non-significant.

With respect to the support modules developed for the AR system, experiments show that named entity recognition and offline animacy detection provide significant improvements to the system, but that PP attachment disambiguation does not.

Named entity recognition

In my work on named entity recognition, I made a decision to divide the NER task into an identification stage and a classification stage, and I let the machine learning method deal only with the classification task. I have suggested that this narrowing of focus may have helped the memory-based learner reach such a good performance, to the point where it even outperforms maximum entropy modelling under some conditions, despite the fact that the latter method has traditionally been more successful at this task.

After testing a set of potentially useful machine learning features for the NER task, I found that the most important ones were the lemma forms of each word in the name ("component lemmas") as well as the presence or absence of the name in a set of gazetteers. I also found that automatic parameter optimization improves the results immensely over the use of default parameter settings or manual optimization.

A technique which turned out to be very useful was to apply a kind of document-centred post-processing in which all occurrences of a name in a particular document are assigned the majority category for those instances, unless the category of a particular name occurrence has been selected by the classifier with a confidence that exceeds a certain threshold.

With respect to evaluation, I concluded that leave-one-out testing is not suitable for evaluation of named entity recognition, and I generalized that conclusion to hold for any classification task in which the classification of an item is influenced by the classification of other items in the test data.

PP attachment disambiguation

In my work on PP attachment disambiguation, I introduced a new, semi-unsupervised way to train a PP attachment disambiguator. This technique makes it possible to make use of large amounts of training data which are not annotated for PP attachment, while at the same time taking advantage of well-established supervised learning methods. Also for this task I compared various machine learning methods, getting much better results for TiMBL and MaxEnt than for SVM, and I tested the use of automatic parameter optimization, which turned out to have either a negative effect or no effect

at all on this task. Additionally, I discussed some issues relating to evaluation, with particular focus on the questionable use of a particular "human-level performance" figure that has been widely cited in the PP attachment disambiguation literature.

Animacy detection

In order to provide the AR system with knowledge about the animacy of Norwegian nouns, I developed two techniques for extracting such information from the World Wide Web. The animacy information produced in this way may be useful also outside the context of anaphora resolution—especially the output of the so-called offline approach, which has the form of a high-precision list of animate nouns. For the anaphora resolution task, I found that recall is more important than precision when it comes to gathering animate nouns, making the kind of automatic extraction techniques described in this thesis particularly suitable for this purpose.

Final words

With the work described in this thesis, I believe I have contributed valuable and much-needed tools and resources to the Norwegian language community. At the same time, I hope to have provided some insight into the development, use, and evaluation of a few important NLP tools.

Bibliography

Aasa, J. (2004). Unsupervised resolution of PP attachment ambiguities in Swedish. Combined C/D level thesis, Stockholm University.

Abney, S. (2002). Bootstrapping. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, pp. 360–367.

Abney, S. (2004). Understanding the Yarowsky Algorithm. *Computational Linguistics 30*(3).

Abney, S., R. E. Schapire, and Y. Singer (1999). Boosting Applied to Tagging and PP Attachment. In P. Fung and J. Zhou (Eds.), *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 38–45.

ACE (2000). Entity detection and tracking – phase 1. http://www.itl.nist.gov/iaui/894.01/tests/ace/phase1/doc/.

Aha, D., D. Kibler, and M. Albert (1991). Instance-based learning algorithms. *Machine Learning* (6), 37–66.

Aizerman, M., E. Braverman, and L. Rozonoer (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control* (25), 821–837.

Aone, C. and S. Bennett (1995). Evaluating automated and manual acquisition of anaphora resolution strategies. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 122–129.

Asher, N. and A. Lascarides (2003). *Logics of Conversation*. Cambridge University Press.

Baayen, R., R. Piepenbrock, and H. van Rijn (1993). The CELEX lexical data base on CD-ROM.

Baldwin, B. (1997). CogNIAC: High precision coreference with limited knowledge and linguistic resources. In *Proceedings of the ACL'97/EACL'97 Workshop on Operational Factors in Practical, Robust Anaphora Resolution*, pp. 38–45.

Beaver, D. (2004). The optimization of discourse anaphora. *Linguistics and Philosophy 27*(1), 3–56.

Bender, O., F. J. Och, and H. Ney (2003). Maximum entropy models for named entity recognition. In *Proceedings of the Seventh Workshop on Computational Language Learning (CoNLL-2003)*, Edmonton, Canada.

Benson, S. J. and J. J. Moré (2001). A limited memory variable metric method for bound constrained minimization, preprint anl/acs-p909-0901. Technical report, Argonne National Library.

Berger, A. L., S. A. Della Pietra, and V. J. Della Pietra (1996). A maximum entropy approach to natural language processing. *Computational Linguistics 22*(1), 39–71.

Berland, M. and E. Charniak (1999). Finding parts in very large corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, Providence, RI, pp. 57–64.

Bick, E. (2004). A named entity recognizer for danish. In *Proceedings of LREC 2004*, pp. 305–308.

Bikel, D., S. Miller, R. Schwartz, and R. Weischedel (1997). Nymble: a high-performance learning name-finder. In *Proceedings of ANLP-97*.

Black, W. J. and A. Vasilakopoulos (2002). Language-independent named entity classification by modified transformation-based learning and by decision tree induction. In *Proceedings of the Sixth Workshop on Computational Language Learning (CoNLL-2002)*, Taipei, Taiwan.

Borthen, K. (2004). Annotation scheme for bredt. version 1.0. Technical report, NTNU, Trondheim, Norway.

Borthen, K., L. G. Johnsen, and C. Johansson (2007). Coding anaphor-antecedent relations; the annotation manual for bredt. In *Proceedings from the first Bergen Workshop on Anaphora Resolution (WAR1)*, pp. 86–111. Cambridge Scholars Publishing.

Borthwick, A. (1999). *A Maximum Entropy Approach to Named Entity Recognition*. Ph. D. thesis, New York University.

Borthwick, A., J. Sterling, E. Agichtein, and R. Grishman (1998). Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In *Proceedings of the 6th Workshop on Very Large Corpora (WVLC 1998)*, pp. 152–160.

Botley, S. (1996). Comparing demonstrative features in three written english genres. In *Proceedings of the Discourse Anaphora and Resolution Colloquium (DAARC96)*, pp. 86–105.

Boyd, A., W. Gegg-Harrison, and D. Byron (2005). Identifying non-referential it: a machine learning approach incorporating linguistically motivated patterns. In *Proceedings of the ACL Workshop on Feature Engineering for Machine Learning in NLP,*, Ann Arbor, pp. 40–47.

Brennan, S. E., M. W. Friedman, and C. J. Pollard (1987). A centering approach to pronouns. In *Proceedings of the 25th Annual Meeting of the ACL*, Stanford, pp. 155–162.

Brill, E. and P. Resnik (1994). A Rule-Based Approach to Prepositional Phrase Attachment Disambiguation. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, Kyoto, Japan.

Buchholz, S. (2002). *Memory-based grammatical relation finding*. Ph. D. thesis, Tilburg University.

Buchholz, S., J. Veenstra, and W. Daelemans (1999). Cascaded grammatical relation assignment. In *EMNLP-VLC'99, the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Burger, J. D., J. C. Henderson, and W. T. Morgan (2002). Statistical named entity recognizer adaptation. In *Proceedings of the Sixth Workshop on Computational Language Learning (CoNLL-2002)*, Taipei, Taiwan.

Burges, C. J. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery 2*(2), 955–974.

Byron, D. K. (2002, 7–12 July 2002). Resolving pronominal reference to abstract entities. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, Penn., pp. 80–87.

Byron, D. K. and J. F. Allen (1998). Resolving demonstrative pronouns in the trains93 corpus. In *New Approaches to Discourse Anaphora: Proceedings of the Second Colloquium on Discourse Anaphora and Anaphor Resolution (DAARC2)*, pp. 68–81.

Canisius, S. and A. van den Bosch (2004). A memory-based shallow parser for spoken dutch. In B. Decadt, G. D. Pauw, and V. Hoste (Eds.), *Selected papers from the Thirteenth Computational Linguistics in the Netherlands Meeting*, Antwerp, Belgium, pp. 31–45.

Caraballo, S. (1999). Automatic acquisition of a hypernym-labelled noun hierarchy from text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, Providence, RI, pp. 120–126.

Carbonell, J. G. and R. D. Brown (1988). Anaphora resolution: A multi-strategy approach. In *Proceedings of the 12th International Conference on Computational Linguistics*, pp. 96–101.

Cardie, C. and K. Wagstaff (1999). Noun phrase coreference as clustering. In *Proceedings of the joint SIGDAT Conference on Empirical Methods in NLP and Very Large Corpora*, University of Maryland, MD, pp. 82–89. Association for Computational Linguistics.

Carreras, X., L. Màrquez, and L. Padró (2003a). Learning a perceptron-based named entity chunker via online recognition feedback. In *Proceedings of the Seventh Workshop on Computational Language Learning (CoNLL-2003)*, Edmonton, Canada.

Carreras, X., L. Màrquez, and L. Padró (2003b). A simple named entity extractor using adaboost. In *Proceedings of the Seventh Workshop on Computational Language Learning (CoNLL-2003)*, Edmonton, Canada.

Carreras, X., L. Márques, and L. Padró (2002). Named entity extraction using adaboost. In *Proceedings of the Sixth Workshop on Computational Language Learning (CoNLL-2002)*, Taipei, Taiwan.

Carter, D. (1987). *Interpreting anaphors in natural language texts*. Chisester: Ellis Horwood ltd.

Charniak, E. (1972). Toward a model of children's story comprehension. Technical Report AI TR-266, Massachusetts Institute of Technology Artificial Intelligence Laboratory.

Charniak, E. (2000). A maximum-entropy-inspired parser. In *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*, Seattle, Washington, pp. 132–139.

Chieu, H. L. and H. T. Ng (2003). Named entity recognition with a maximum entropy approach. In *Proceedings of the Seventh Workshop on Computational Language Learning (CoNLL-2003)*, Edmonton, Canada.

Chinchor, N. (1997). Overview of MUC-7. In *Proceedings of MUC-7*.

Chinchor, N., E. Brown, L. Ferro, and P. Robinson (1999). 1999 named entity recognition task definition. MITRE and SAIC.

Chomsky, N. (1981). *Lectures on Government and Binding*. Dordrecht: Foris.

Chomsky, N. (1986). *Barriers*. Cambridge, Mass.: MIT Press.

Church, K. and W. Gale (1991). A comparison of the enhanced good-turing and deleted estimation methods for estimating probabilities of english bigrams. *Computer Speech and Language 5*, 19–54.

Collins, M. and J. Brooks (1995). Prepositional attachment through a backed-off model. In D. Yarowsky and K. Church (Eds.), *Proceedings of the Third Workshop on Very Large Corpora*, Somerset, New Jersey, pp. 27–38. Association for Computational Linguistics.

Connolly, D., J. D. Burger, and D. S. Day (1997). A machine learning approach to anaphoric reference. *New Methods in Language Processing*, 133–144.

Cost, S. and S. Salzberg (1993). A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning 10*, 57–78.

Cover, T. and P. Hart (1967). Nearest neighbour pattern classification. *Institute of Electrical and Electronics Engineers Transactions on Information Theory 13*, 21–27.

Curran, J. and S. Clark (2003a). Language independent ner using a maximum entropy tagger. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*.

Curran, J. R. and S. Clark (2003b). Investigating gis and smoothing for maximum entropy taggers. In *Proceedings of the 11th Annual Meeting of the European Chapter of the Association for Computational Linguistics (EACL'03)*, Budapest, Hungary, pp. 91–98.

Daelemans, W. (1995). Memory-based lexical acquisition and processing. In P. Steffens (Ed.), *Machine Translation and the Lexicon*, Volume 898 of *Lecture Notes in Artificial Intelligence*, pp. 85–98. Berlin: Springer-Verlag.

Daelemans, W. (1996). Abstraction considered harmful: Lazy learning of language processing. In J. van den Herik and T. Weijters (Eds.), *Benelearn-96. Proceedings of the 6th Belgian-Dutch Conference on Machine Learning*, Maastricht, The Netherlands.

Daelemans, W. and A. van den Bosch (2005). *Memory-Based Language Processing*. Studies in Natural Language Processing. Cambridge University Press.

Daelemans, W., A. van den Bosch, and J. Zavrel (1999). Forgetting exceptions is harmful in language learning. *Machine Learning 34* (1-3), 11–41.

Daelemans, W., J. Zavrel, K. van der Sloot, and A. van den Bosch (2004). TiMBL: Tilburg Memory-Based Learner Reference Guide. Version 5.1. Technical Report 04-02, ILK.

Dagan, I. and A. Itai (1990). Automatic processing of large corpora for the resolution of anaphora references. In *Proceedings of the 13th conference on Computational linguistics*, Volume 3, Helsinki, Finland, pp. 330–332.

Dalianis, H. (2000). Swesum - a text summarizer for swedish. Technical report, KTH.

Darroch, J. and D. Ratcliff (1972). Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics 43*, 1470–1480.

Davidson, D. (1980). *Essays on Actions and Events*. Oxford University Press.

Decadt, B., J. Duchateau, W. Daelemans, and P. Wambacq (2002). Memory-based phoneme-to-grapheme conversion. In M. Theune, A. Nijholt, and H. Hondrop (Eds.), *Computational Linguistics in the Netherlands 2001. Selected Papers from the Twelfth CLIN Meeting, Amsterdam*, pp. 47–61. Rodopi.

Decadt, B., V. Hoste, W. Daelemans, and A. van den Bosch (2004). GAMBL, genetic algorithm optimization of memory-based WSD. In R. Mihalcea and P. Edmonds (Eds.), *Proceedings of the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (Senseval-3)*, Barcelona, Spain, pp. 108–112.

Della Pietra, S., V. J. Della Pietra, and J. D. Lafferty (1997). Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19* (4), 380–393.

Denber, M. (1998). Automatic resolution of anaphora in english. Eastman Kodak Co.

Devijver, P. and J. Kittler (1980). On the edited nearest neighbour rule. In *Proceedings of the Fifth International Conference on Pattern Recognition*. The Institute of Electrical and Electronics Engineers.

Dudani, S. A. (1976). The distance-weighted k-nearest neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics SMC-6*, 325–327.

Dyvik, H. (2002). Translations as semantic mirrors: From parallel corpus to wordnet. Paper presented at The 23rd International Conference on English Language Research on Computerized Corpora of Modern and Medieval English (ICAME 2002), Göteborg 22-26 May 2002.

Dyvik, H. (2003). Translations as a semantic knowledge source. Unpublished draft.

Eckert, M. and M. Strube (2000). Dialogue acts, synchronising units and anaphora resolution. In *Journal of Semantics*, Volume 17, pp. 51–89.

Eugenio, B. D. (1990). Centering Theory and the Italian pronominal system. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING 90)*, Helsinki, pp. 270–275.

Eugenio, B. D. (1998). Centering in Italian. In A. K. J. Marilyn A. Walker and E. F. Prince (Eds.), *Centering Theory in Discourse*, pp. 115–138. Oxford University Press.

Evans, R. (2001). Applying machine learning toward an automatic classification of it. *Literary and Linguistic Computing 16*(1), 45–57.

Evans, R. and C. Orasan (2000). Improving anaphora resolution by identifying animate entities in texts. In *Proceedings of the Discourse Anaphora and Reference Resolution Conference (DAARC2000)*, Lancaster, UK, pp. 154–162.

Faarlund, J. T., S. Lie, and K. I. Vannebo (1997). *Norsk referansegrammatikk*. Universitetsforlaget.

Fellbaum, C. (Ed.) (1998). *WordNet. An Electronic Lexical Database*. The MIT Press.

Fisher, F., S. Soderland, J. McCarthy, F. Feng, and W. Lehnert (1995). Description of the UMASS system as used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 127–140.

Florian, R., A. Ittycheriah, H. Jing, and T. Zhang (2003). Named entity recognition through classifier combination. In *Proceedings of the Seventh Workshop on Computational Language Learning (CoNLL-2003)*, Edmonton, Canada.

Franz, A. (1996). *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*, Volume 1040 of *Lecture Notes in Artificial Intelligence*, Chapter Learning PP attachment from corpus statistics, pp. 188–202. New York: Springer-Verlag.

Fraurud, K. (1992). *Processing Noun Phrases in Natural Discourse*. Ph. D. thesis, Stockholm University.

Frazier, L. (1979). *On Comprehending Sentences: Syntactic Parsing Strategies*. Ph. D. thesis, University of Connecticut.

Gale, W., K. Church, and D. Yarowsky (1992a). A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 415–439.

Gale, W., K. Church, and D. Yarowsky (1992b). One sense per discourse. In *Proceedings of the Fourth DARPA Speech and Natural Language Workshop*, pp. 233–237.

Ge, N., J. Hale, and E. Charniak (1998). A statistical approach to anaphora resolution. In *Proceedings of the Sixth Workshop on Very Large Corpora*.

Goldman, S. and Y. Zhou (2000). Enhancing supervised learning with unlabeled data. In *Proceedings of ICML*, pp. 327–334.

Grimshaw, J. (1990). *Argument Structure*. Cambridge, Massachusetts: MIT Press.

Grishman, R. and B. Sundheim (1996). Message understanding conference 6: A brief history. In *Proceedings of the 16th International Conference on Computational Linguistics*, Copenhagen.

Grosz, B. J. (1977). The representation and use of focus in dialogue understanding. Technical Report 151, SRI International, 333 Ravenswood Ave, Menlo Park, Ca. 94025.

Grosz, B. J. (1981). *Elements of Discourse Understanding*, Chapter Focusing and description in natural language dialogues, pp. 85–105. Cambridge, England: Cambridge University Press.

Grosz, B. J., A. K. Joshi, and S. Weinstein (1983). Providing a unified account of definite noun phrases in discourse. In *Proceedings of the 21st Annual Meeting of the ACL*, pp. 44–50. Association of Computational Linguistics.

Grosz, B. J., A. K. Joshi, and S. Weinstein (1995). Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics 21*(2), 203–225.

Gundel, J. (2002). Information structure and the use of cleft sentences in English and Norwegian. In H. Hasselgård, S. Johansson, B. Behrens, and C. Fabricius-Hansen (Eds.), *Information Structure in a Cross-linguistic Perspective*, Volume 39 of *Language and Computers*, pp. 113–129. Amsterdam: Rodopi.

Haaland, Å. (2008). *A Maximum Entropy Approach to Proper Name Classification for Norwegian*. Ph. D. thesis, University of Oslo.

Hagen, K. (2003). Automatisk sammenslåing av navn. In H. Holmboe (Ed.), *Nordisk Sprogteknologi 2002. Årbok for Nordisk Sprogteknologisk Forskningsprogram 2000-2004*, pp. 351–356. Museum Tusculanums Forlag.

Hagen, K., J. B. Johannessen, and A. Nøklestad (2000a). A Constraint-Based Tagger for Norwegian. In C.-E. Lindberg and S. Nordahl Lund (Eds.), *17th Scandinavian Conference of Linguistics*, Volume I of *Odense Working Papers in Language and Communication*, Odense.

Hagen, K., J. B. Johannessen, and A. Nøklestad (2000b). A Web-Based Advanced and User Friendly System: The Oslo Corpus of Tagged Norwegian Texts. In M. Gavrilidou, G. Carayannis, S. Markantonatou, S. Piperidis, and G. Stainhaouer (Eds.), *Proceedings of the Second International Conference on Language Resources and Evaluation*, Athens, Greece.

Hale, J. and E. Charniak (1998). Getting useful gender statistics from english text. Technical Report CS-98-06, Brown University.

Haltrup, D. (2003). Automatisk navnegenkendelse for dansk. In H. Holmboe (Ed.), *Nordisk Sprogteknologi 2002. Årbok for Nordisk Sprogteknologisk Forskningsprogram 2000-2004*, pp. 357–360. Museum Tusculanums Forlag.

Hammerton, J. (2003). Named entity recognition with long short-term memory. In *Proceedings of the Seventh Workshop on Computational Language Learning (CoNLL-2003)*, Edmonton, Canada.

Hardt, D. (2004). Dynamic centering. In *Proceedings of Reference Resolution and its Applications, Workshop at ACL 2004*, Barcelona.

Hassel, M. (2000). Pronominal resolution in automatic text summarisation. Master's thesis, Department of Computer and Systems Sciences, Stockholm University.

Hastie, T., R. Tibshirani, and J. Friedman (2001). *The Elements of Statistical Learning. Data Mining, Inference, and Prediction.* Springer Series in Statistics. New York: Springer.

Hearst, M. (1992). Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*, Nantes, France.

Hellan, L. (1988). *Anaphora in Norwegian and the theory of grammar.* Dordrecht: Foris.

Hendrickx, I., V. Hoste, and W. Daelemans (2008). Semantic and syntactic features for dutch coreference resolution. In A. Gelbukh (Ed.), *Proceedings of the CICLing-2008 conference*, Volume 4919 of *Lecture Notes in Computer Science*, Berlin, pp. 351–361. Springer Verlag.

Hendrickx, I. and A. van den Bosch (2003). Memory-based one-step named-entity recognition: Effects of seed list features, classifier stacking, and unannotated data. In W. Daelemans and M. Osborne (Eds.), *Proceedings of CoNLL-2003, the Seventh Conference on Natural Language Learning*, Edmonton, Canada, pp. 176–179.

Hindle, D. and M. Rooth (1993). Structural ambiguity and lexical relations. *Computational Linguistics 19*(1), 103–120.

Hobbs, J. (1978). Resolving pronoun references. *Lingua 44*, 339–352.

Hobbs, J., M. Stickel, D. Appelt, and P. Martin (1993). Interpretation as abduction. *Artificial Intelligencde 63*, 69–142.

Hobbs, J. R., M. Stickel, P. Martin, and D. Edwards (1988). Interpretation as abduction. In *Proceedings of the 26th annual meeting on Association for Computational Linguistics*, Buffalo, New York, pp. 95–103.

Hofland, K. (2000). A self-expanding corpus based on newspapers on the web. In *Proceedings of the Second International Language Resources and Evaluation Conference*, Paris.

Holen, G. I. (2006). Automatic anaphora resolution for norwegian (ARN). Cand. philol. thesis in language, logic and information, University of Oslo.

Hoste, V. (2005). *Optimization Issues in Machine Learning of Coreference Resolution.* Ph. D. thesis, University of Antwerp.

Hoste, V., W. Daelemans, I. Hendrickx, and A. van den Bosch (2002). Evaluating the results of a memory-based word-expert approach to unrestricted word sense disambiguation. In *Proceedings of the Workshop on word sense disambiguation: Recent successes and future directions*, Philadelphia, PA, pp. 95–101.

Hoste, V. and A. van den Bosch (2007). A modular approach to learning dutch co-reference. In C. Johansson (Ed.), *Proceedings of the Workshop on Anaphora Resolution, Mjølfjell, Norway, September 28-30, 2005.*

Iida, R., K. Inui, H. Takamura, and Y. Matsumoto (2003). Incorporating contextual cues in trainable models for coreference resolution. In *Proceedings of the EACL Workshop on The Computational Treatment of Anaphora.*

Isozaki, H. and H. Kazawa (2002). Efficient Support Vector Classifiers for Named Entity Recognition. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING-2002)*, pp. 390–396.

Jackendoff, R. (1990). *Semantic Structures*. Cambridge, Massachusetts: MIT Press.

Jansche, M. (2002). Named entity extraction with conditional markov models and classifiers. In *Proceedings of the Sixth Workshop on Computational Language Learning (CoNLL-2002)*, Taipei, Taiwan.

Jaynes, E. (1983). *Papers on Probability, Statistics, and Statistical Physics*. D. Reidel Publishing Company.

Joachims, T. (1999). Making large-Scale SVM Learning Practical. In B. Schölkopf, C. Burges, and A. Smola (Eds.), *Advances in Kernel Methods - Support Vector Learning*. MIT-Press.

Johannessen, J. B., K. Hagen, Å. Haaland, A. B. Jónsdottir, A. Nøklestad, D. Kokkinakis, P. Meurer, E. Bick, and D. Haltrup (2005). Named entity recognition for the mainland scandinavian languages. *Literary and Linguistic Computing 20*(1), 91–102.

Johannessen, J. B. and H. Hauglin (1998). An automatic analysis of norwegian compounds. In T. Haukioja (Ed.), *Papers from the 16th Scandinavian conference of lingvistics*, Turku/Åbo. University of Turku.

Johannessen, J. B., P. Meurer, and K. Hagen (2003). Recognising word strings as names. In *Proceedings of NoDaLiDa (Nordiske datalingvistikkdager) 2003*, Reykjavík, Iceland.

Johansson, C. and L. G. Johnsen (2006). Analogical modeling with bias – allowing feedback and centering. *Special issue on Advances in Natural Language Processing, Research on Computing Science*. National Polytechnic Institute of Mexico, Mexico City.

Joshi, A. K. and S. Kuhn (1979). Centered logic: The role of entity centered sentence representation in natural language inferencing. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

Joshi, A. K. and S. Weinstein (1981). Control of inference: Role of some aspects of discourse structure - centering. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 385–387.

Jónsdóttir, A. B. (2003). Arner, what kind of name is that? - An Automatic Rule-based Named Entity Recognizer for Norwegian. Cand. philol. thesis in language, logic and information, University of Oslo.

Kameyama, M. (1986). A property-sharing constraint in centering. In *Proceedings of the 24th ACL*, New York, pp. 200–206.

Kameyama, M. (1993). Intrasentential centering. In *Proceedings of the Workshop on Centering*. University of Pennsylvania.

Kameyama, M. (1998). Intrasentential centering: A case study. In A. K. J. Marilyn A. Walker and E. F. Prince (Eds.), *Centering in Discourse*, pp. 89–114. Oxford University Press.

Kamp, H. and U. Reyle (1993). *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic, and Discourse Representation Theory*. Boston: Kluwer Academic.

Karlsson, F., A. Voutilainen, J. Heikkilä, and A. Anttila (1995). *Constraint Grammar: A Language-Independent Framework for Parsing Unrestricted TExt*. Mouton de Gruyter.

Karttunen, L. (1976). *Syntax and Semantics*, Volume 7, Chapter Discourse Referents. New York: Academic Press.

Katz, S. (1987). Estimation of probabilities from sparse data for the language model component of a speech recognizer. *EEE Trans. on Acoustics, Speech and Signal Processing 35*(3), 400–401.

Kazama, J. (2004). *Improving Maximum Entropy Natural Language Processing by Uncertainty-Aware Extensions and Unsupervised Learning*. Ph. D. thesis, University of Tokyo.

Keenan, E. and B. Comrie (1977). Noun phrase accessibility and universal grammar. *Linguistic Inquiry 8*, 62–100.

Kehler, A. (1997). Current theories of centering for pronoun interpretation: A critical evaluation. *Computational Linguistics 23*(3), 467–475.

Kehler, A. (2000). *Speech and Language Processing*, Chapter Pragmatics, Chapter 18. Upper Saddle River, NJ: Prentice Hall.

Kehler, A. (2002). *Coherence, Reference, and the Theory of Grammar*. Stanford, CA: CSLI Publications.

Kennedy, C. and B. Boguraev (1996). Anaphora for everyone: Pronominal anaphora resolution without a parser. In *Proceedings of the 16th International Conference on Computational Linguistics*.

Kimball, J. (1973). Seven principles of surface structure parsing in natural language. *Cognition 2*(1), 15–47.

Klein, D., J. Smarr, H. Nguyen, and C. D. Manning (2003). Named entity recognition with character-level models. In *Proceedings of the Seventh Workshop on Computational Language Learning (CoNLL-2003)*, Edmonton, Canada.

Klenner, M. (2007). Enforcing consistency on coreference sets. In *Proceedings of RANLP '07*.

Klenner, M. and Étienne Ailloud (2008). Enhancing coreference clustering. In C. Johansson (Ed.), *Proceedings of the Second Workshop on Anaphora Resolution (WARII)*.

Kokkinakis, D. (2000). Supervised pp-attachment for swedish (combining unsupervised and supervised training data). *Nordic Journal of Linguistics 3*.

Kokkinakis, D. (2001). Design, implementation and evaluation of named-entity recognizer for swedish. Technical Report GU-ISS-01-2, Språkdata, University of Gothenburg.

Kokkinakis, D. (2002). Namnigenkänning på svenska. In H. Holmboe (Ed.), *Nordisk Sprogteknologi 2001. Årbok for Nordisk Sprogteknologisk Forskningsprogram 2000-2004*, pp. 167–171. Museum Tusculanums Forlag.

Kokkinakis, D. (2003). Swedish ner in the Nomen Nescio project. In H. Holmboe (Ed.), *Nordisk Sprogteknologi 2002. Årbok for Nordisk Sprogteknologisk Forskningsprogram 2000-2004*, pp. 379–398. Museum Tusculanums Forlag.

Kokkinakis, D. (2004). Reducing the effect of name explosion. In L. Guthrie, R. Basili, E. Hajicova, and F. Jelinek (Eds.), *Proceedings of the LREC Workshop: Beyond Named Entity Recognition: Semantic Labelling for NLP Tasks*, Lisbon, Portugal, pp. 1–6.

Kudo, T. and Y. Matsumoto (2000). Japanese dependency structure analysis based on support vector machines. In *Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 18–25.

Kudo, T. and Y. Matsumoto (2001). Chunking with support vector machines. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on language technologies*, Pittsburgh, Pennsylvania.

Lappin, S. (2005). A sequenced model of anaphora and ellipsis resolution. In A. Branco, T. McEnery, and R. Mitkov (Eds.), *Anaphora Processing. Linguistic, cognitive and computational modelling*, Volume 263 of *Current Issues in Linguistic Theory*, pp. 3–16. John Benjamins Publishing Company.

Lappin, S. and H. J. Leass (1994). An algorithm for pronominal anaphora resolution. *Computational Linguistics 20*(4), 535–561.

Lappin, S. and M. McCord (1990). A syntactic filter on pronominal anaphora in slot grammar. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pp. 135–142.

Lech, T. C. and K. de Smedt (2007). Ontology extraction for coreference chaining. In C. Johansson (Ed.), *Proceedings of the Workshop on Anaphora Resolution, Mjølfjell, Norway, September 28-30, 2005.*

Lendvai, P., A. van den Bosch, and E. Krahmer (2003). Memory-based disfluency chunking. In *Proceedings of DISS'03, Disfluency in Spontaneous Speech Workshop*, Göteborg University, Sweden, pp. 63–66.

Lendvai, P., A. van den Bosch, E. Krahmer, and S. Canisius (2004). Memory-based robust interpretation of recognised speech. In *Proceedings of the 9th International Conference on "Speech and Computer", SPECOM'04*, St. Petersburg, Russia, pp. 415–422.

Levenshtein, V. I. (1965). Binary codes capable of correcting insertions, deletions, and reversals. *Doklady Akademii Nauk SSSR 163*(4), 845–848. Original article in Russian. English translation in Soviet Physics Doklady, 10(8):707–710, 1966.

Lyse, G. I. (2003). Fra speilmetoden til automatisk ekstrahering av et betydningstagget korpus for wsd-formål. Cand. philol. thesis in linguistics, University of Bergen.

Lødrup, H. (2007). Norwegian anaphors without visible binders. *Jounal of Germanic Linguistics 19*(1), 1–22.

Malouf, R. (2002a). A comparison of algorithms for maximum entropy parameter estimation. In *Proccedings of the Sixth Conference on Natural Language Learning*, pp. 49–55.

Malouf, R. (2002b). Markov models for language-independent named entity recognition. In *Proceedings of the Sixth Workshop on Computational Language Learning (CoNLL-2002)*, Taipei, Taiwan.

Marcus, M. P., B. Santorini, and M. A. Marcinkiewicz (1993). Building a large annotated corpus of english: The penn treebank. *Computational Linguistics* (2), 313–330.

Markert, K. and M. Nissim (2005). Comparing knowledge sources for nominal anaphora resolution. *Computational Linguistics 31*(3), 367–402.

Marsi, E., B. Busser, W. Daelemans, V. Hoste, M. Reynaert, and A. van den Bosch (2002). Combining information sources for memory-based pitch accent placement. In *Proceedings of the International Conference on Spoken Language Processing*, pp. 1273–1276.

Marsi, E., P.-A. Coppen, C. Gussenhoven, and T. Rietveld (1997). *Progress in Speech Synthesis*, Chapter Prosodic and intonational domains in speech synthesis, pp. 477–493. New York: Springer-Verlag.

Marsi, E., A. van den Bosch, and A. Soudi (2005). Memory-based morphological analysis generation and part-of-speech tagging of arabic. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, Ann Arbor, MI.

Mayfield, J., P. McNamee, and C. Piatko (2003). Named entity recognition using hundreds of thousands of features. In *Proceedings of the Seventh Workshop on Computational Language Learning (CoNLL-2003)*, Edmonton, Canada.

McCallum, A. and W. Li (2003). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*.

McCarthy, J. F. (1996). *A Trainable Approach to Coreference Resolution for Information Extraction*. Ph. D. thesis, University of Massachusetts Amherst, Department of Computer Science.

McCarthy, J. F. and W. G. Lehnert (1995). Using decision trees for coreference resolution. In C. Mellish (Ed.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence*, pp. 1050–1055.

McDonald, D. (1996). *Corpus Processing for Lexical Acquisition*, Chapter Internal and external evidence in the identification and semantic categorization of proper names, pp. 21–39. Cambridge, Mass: MIT Press.

McNamee, P. and J. Mayfield (2002). Entity extraction without language-specific resources. In *Proceedings of the Sixth Workshop on Computational Language Learning (CoNLL-2002)*, Taipei, Taiwan.

Merlo, P. and E. E. Ferrer (2006). The notion of argument in prepositional phrase attachment. *Computational Linguistics 32*(3), 341–377.

Meulder, F. D. and W. Daelemans (2003). Memory-based named entity recognition using unannotated data. In W. Daelemans and M. Osborne (Eds.), *Proceedings of CoNLL-2003, the Seventh Conference on Natural Language Learning*, Edmonton, Canada, pp. 208–211.

Meyer, I. (2001). *Recent Advances in Computational Terminology*, Chapter Extracting knowledge-rich contexts for terminography, pp. 279–301. Amsterdam: John Benjamins.

Mihalcea, R. (2002). Instance based learning with automatic feature selection applied to word sense disambiguation. In *Proceedings of COLING-2002*, pp. 660–666.

Mikheev, A. (1998). Feature lattices for maximum entropy modelling. In *Proceedings of the 36th Conference of the Association for Computational Linguistics*, Montreal, Quebec, pp. 848–854.

Mikheev, A. (2000). Document centered approach to text normalization. In *Proceedings of SIGIR'2000*, pp. 136–143.

Mikheev, A. (2002). Periods, capitalized words, etc. *Computational Linguistics 28*(3), 289–318.

Mikheev, A., C. Grover, and M. Moens (1998). Description of the ltg system used for MUC-7. In *Proceedings of the 7th Message Understanding Conference (MUC-7)*.

Mikheev, A., M. Moens, and C. Grover (1999). Named entity recognition without gazetteers. Bergen, Norway. EACL.

Miltsakaki, E. (1999). Locating topics in text processing. In *Computational Linguistics in the Netherlands: Selected Papers from the Tenth CLIN Meeting (CLIN '99)*, Utrecht, pp. 127–138.

Minkov, E., R. C. Wang, and W. W. Cohen (2005). Extracting personal names from email: Applying named entity recognition to informal text. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing 2005*.

Minsky, M. and S. Papert (1969). *Perceptrons. An Introduction to Computational Geometry*. Cambridge, MA: MIT Press.

Mitchell, B. and R. Gaizauskas (2002). A Comparison of Machine Learning Algorithms for Prepositional Phrase Attachment. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002)*, Las Palmas, Gran Canaria, Spain, pp. 2082–2087.

Mitchell, T. (1997). *Machine Learning*. New York: McGraw-Hill.

Mitkov, R. (1998). Robust pronoun resolution with limited knowledge. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING'98/ACL'98)*, Montreal, Canada, pp. 869–875.

Mitkov, R. (2002). *Anaphora Resolution*. London: Longman (Pearson Education).

Montague, R. (1974). *Formal Philosophy*, Chapter The Proper Treatment of Quantification in Ordinary English, pp. 188–221. New Haven, Connecticut: Yale University Press.

Nararretta, C. (2004). Resolving individual and abstract anaphora in text and dialogues. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-2004)*, Geneva, Switzerland.

Navarretta, C. (2002). Combining information structure and centering-based models of salience for resolving intersentential pronominal anaphora. In A. Branco, T. McEnery, and R. Mitkov (Eds.), *Proceedings of DAARC 2002 - 4h Discourse Anaphora and Anaphora Resolution Colloquium*, pp. 135–140. Ediçoes Colibri.

Ng, V. (2005). Machine learning for coreference resolution: From local classification to global ranking. In *Proceedings of the 43rd Meeting of the ACL*, Ann Arbor, pp. 157–164. Association for Computational Linguistics.

Ng, V. and C. Cardie (2002a). Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In *Proceedings of the 19th international conference on Computational linguistics*, Volume 1, Taipei, Taiwan.

Ng, V. and C. Cardie (2002b). Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 104–111.

Ng, V. and C. Cardie (2003a). Bootstrapping coreference classifiers with multiple machine learning algorithms. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pp. 113–120.

Ng, V. and C. Cardie (2003b). Weakly supervised natural language learning without redundant views. In *Proceedings of HLT-NAACL*.

Nygaard, L. (2006). Frå ordbok til ordnett. Cand. philol. thesis in linguistics, University of Oslo.

Nøklestad, A. (2004). Memory-based Classification of Proper Names in Norwegian. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC-2004)*, Volume II, pp. 439–442.

Nøklestad, A. (2005). Semi-unsupervised pp attachment disambiguation for norwegian. *Archives of Control Sciences, Special issue on Human Language Technologies as a challenge for Computer Science and Linguistics. Part I 15*.

Nøklestad, A. and C. Johansson (2005). Detecting reference chains in norwegian. In S. Werner (Ed.), *Proceedings of the 15th Nodalida Conference*, pp. 156–162.

Nøklestad, A., C. Johansson, and A. van den Bosch (2004). Using Z-scores to improve memory based anaphora resolution. Poster presented at the 15th Meeting of Computational Linguistics in the Netherlands.

Nøklestad, A., Ø. Reigem, and C. Johansson (2006). Developing a re-usable web-demonstrator for automatic anaphora resolution with support for manual editing of coreference chains. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, pp. 1161–1166.

Okanohara, D., Y. Miyao, Y. Tsuruoka, and J. Tsujii (2006). Improving the scalability of semi-markov conditional random fields for named entity recognition. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*, Sydney, pp. 465–472. Association for Computational Linguistics.

Olteanu, M. and D. Moldovan (2005). Pp-attachment disambiguation using large context. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, Vancouver, pp. 273–280.

Orasan, C. and R. Evans (2001). Learning to identify animate references. In *Proceedings of the Workshop Computational Natural Language Learning 2001 (CoNLL-2001)*, Toulouse.

Pantel, P. and D. Lin (2000). An Unsupervised Approach to Prepositional Phrase Attachment using Contextually Similar Words. In *Proceedings of Association for Computational Linguistics (ACL-00)*, Hong Kong, pp. 101–108.

Poesio, M., R. Stevenson, B. D. Eugenio, and J. Hitzeman (2004). Centering: A parametric theory and its instantiations. *Computational Linguistics 30*(3).

Pradhan, S., K. Hacioglu, V. Krugler, W. Ward, J. H. Martin, and D. Jurafsky (2005). Support vector learning for semantic argument classification. *Machine Learning 60*(1), 11–39.

Prince, A. and P. Smolensky (1993). Optimality theory: Constraint interaction in generative grammar. technical report 2. Technical report, Rutgers University Center for Cognitive Science.

Prince, E. (1981). *Radical Pragmatics*, Chapter Towards a taxonomy of given-new information, pp. 223–255. New York: Academic Press.

Prince, E. (1992). *Discourse Description: Diverse Linguistic Analyses of a Fund-Raising Text*, Chapter The ZPG letter: Subjects, definiteness, and information-status, pp. 295–325. Amsterdam: John Benjamins.

Quinlan, J. (1993). *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Ratnaparkhi, A. (1996). A maximum entropy part-of-speech tagger. In *Proceedings of the Empirical Methods in Natural Language Processing Conference (EMNLP)*, University of Pennsylvania.

Ratnaparkhi, A. (1998). Statistical models for unsupervised prepositional phrase attachement. In *COLING-ACL*, pp. 1079–1085.

Ratnaparkhi, A. (1999). Learning to parse natural language with maximum entropy models. *Machine Learning 34*(1–3), 151–175.

Ratnaparkhi, A., J. Reynar, and S. Roukos (1994). A Maximum Entropy Model for Prepositional Phrase Attachment. In *Proceedings of the ARPA Workshop on Human Language Technology*. Morgan Kaufmann.

Rich, E. and S. LuperFoy (1988). An architecture for anaphora resolution. In *Proceedings of the Second Conference on Applied Natural Language Processing*, Austin, Texas.

Rijsbergen, C. J. V. (1979). *Information Retrieval*. Newton, MA: Butterworth-Heinemann.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review 65*(6), 386–408.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Volume Volume 1: Foundations, Chapter Learning internal representations by error propagation. Cambridge, MA: MIT Press.

Rømcke, A. (2008). Named entity recognition using the web: The truth is out there. Master's thesis, University of Bergen.

Sag, I. A., T. Baldwin, F. Bond, A. Copestake, and D. Flickinger (2002). Multiword expressions: A pain in the neck for NLP. In *Proc. of the 3rd International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2002)*, Mexico City, Mexico, pp. 1–15.

Sapir, E. (1921). *Language*. New York: Harcourt Brace Jovanovich.

Schütze, H. (1997). *Ambiguity resolution in language learning*. Standford CA: CSLI Publications.

Sekine, S., R. Grishman, and H. Shinnou (1998). A decision tree method for finding and classifying names in japanese texts. In E. Charniak (Ed.), *Proceedings of the Sixth Workshop on Very Large Corpora*, Montréal, Canada.

Sekine, S. and H. Isahara (2000). Irex: Ir and ie evaluation project in japanese. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, Athens, pp. 1475–1480.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal 27*, 379–423 and 623–656.

Shen, D., J. Zhang, J. Su, G. Zhou, and C.-L. Tan (2004). Multi-criteria-based active learning for named entity recognition. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004)*, pp. 590–597.

Sidner, C. L. (1979). *Towards a Computational Theory of Definite Anaphora Comprehension in English Discourse*. Ph. D. thesis, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Soon, W., H. Ng, and D. Lim (2001). A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics 27*(4), 521–544.

Sparck Jones, K. and P. Willett (1997). *Readings in Information Retrieval*, Chapter 1: Overall Introduction. Morgan Kaufmann Publishers, Inc.

Stanfill, C. and D. Waltz (1986). Toward memory-based reasoning. *Communications of the ACM 29*(12), 1213–1228.

Steedman, M., M. Osborne, A. Sarkar, S. Clark, R. Hwa, J. Hockenmaier, P. Ruhlen, S. Baker, and J. Crim (2003). Bootstrapping statistical parsers from small datasets. In *Proceedings of the EACL*.

Stetina, J. and M. Nagao (1997). Corpus Based PP-Attachment Ambiguity Resolution with a Semantic Dictionary. In J. Zhou and K. Church (Eds.), *Proceedings of the 5th Workshop on Very Large Corpora*, China & Hong Kong, pp. 66–80.

Strube, M. (1998). Never look back: An alternative to centering. In *Proceedings of ACL '98*, pp. 1251–1257.

Strube, M. and U. Hahn (1996). Functional centering. In A. Joshi and M. Palmer (Eds.), *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, San Francisco, pp. 270–277. Morgan Kaufmann Publishers.

Strube, M. and U. Hahn (1999). Functional centering. *Computational Linguistics 25*(3).

Strube, M. and C. Müller (2003). A machine learning approach to pronoun resolution in spoken dialogue. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pp. 168–175.

Strube, M., S. Rapp, and C. Müller (2002). The influence of minimum edit distance on reference resolution. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, Philadelphia, Pa., pp. 312–319.

Stuckard, R. (2001). Design and enhanced evaluation of a robust anaphor resolution algorithm. *Computational Linguistics 27*(4), 473–506.

Sveen, A. (1996). *Norwegian Impersonal Actives and the Unaccusative Hypothesis*. Ph. D. thesis, University of Oslo.

Tapanainen, P. (1996). *The Constraint Grammar Parser CG-2*. Number 27 in Publications. Helsinki: University of Helsinki, Department of Linguistics.

Tapanainen, P. and T. Järvinen (1997). A non-projective dependency parser. In *Proceedings of the 5th Conference of Applied Natural Language Processing (ANLP-5)*, Washington, DC, USA, pp. 64–71.

Tetreault, J. (1999). Analysis of syntax-based pronoun resolution methods. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pp. 602–605.

Tetreault, J. R. (2001). A corpus-based evaluation of centering and pronoun resolution. *Computational Linguistics 27*(4), 507–520.

Thunes, M. (2003). Evaluating thesaurus entries derived from translational features. In *Proceedings of NoDaLiDa (Nordiske datalingvistikkdager) 2003*, Reykjavík, Iceland.

Tjong Kim Sang, E. F. (2002a). Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2002, the Sixth Conference on Natural Language Learning*, Taipei, Taiwan, pp. 155–158.

Tjong Kim Sang, E. F. (2002b). Memory-based named entity recognition. In *Proceedings of CoNLL-2002, the Sixth Conference on Natural Language Learning*, Taipei, Taiwan, pp. 203–206.

Tjong Kim Sang, E. F. and F. De Meulder (2003). Introduction to the conll-2003 shared task: Language-independent named entity recognition. In W. Daelemans and M. Osborne (Eds.), *Proceedings of CoNLL-2003, the Seventh Conference on Natural Language Learning*, Edmonton, Canada, pp. 142–147.

van de Cruys, T. (2005). Semantic clustering in dutch. In *Proceedings of the Sixteenth Computational Linguistics in the Netherlands (CLIN)*, pp. 17–32.

van den Bosch, A. (2004). Wrapped progressive sampling search for optimizing learning algorithm parameters. In R. Verbrugge, N. Taatgen, and L. Schomaker (Eds.), *Proceedings of the 16th Belgian-Dutch Conference on Artificial Intelligence*, Groningen, The Netherlands.

van den Bosch, A. (2005). Memory-based understanding of user utterances in a spoken dialogue system: Effects of feature selection and co-learning. In *Workshop Proceedings of the 6th International Conference on Case-Based Reasoning*, Chicago, pp. 85–94.

van den Bosch, A., S. Canisius, W. Daelemans, I. Hendrickx, and E. T. K. Sang (2004). Memory-based semantic role labeling: Optimizing features, algorithm, and output. In H. Ng and E. Riloff (Eds.), *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*, Boston, MA.

van Halteren, H., J. Zavrel, and W. Daelemans (2001). Improving accuracy in word class tagging through the combination of machine learning systems. *Computational Linguistics 27*(2), 199–230.

van Herwijnen, O., J. Terken, A. van den Bosch, and E. Marsi (2003). Learning PP attachment for filtering prosodic phrasing. In *Proceedings of EACL 2003, 10th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 139–146.

Vanschoenwinkel, B. and B. Manderick (2003). A weighted polynomial information gain kernel for resolving prepositional phrase attachment ambiguities with support vector machines. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pp. 133–140.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. New York: Springer-Verlag.

Veenstra, J., A. van den Bosch, S. Buchholz, W. Daelemans, and J. Zavrel (2000). Memory-based word sense disambiguation. *Computers and the Humanities*.

Velldal, E. (2003). Modeling Word Senses With Fuzzy Clustering. Cand. philol. thesis in language, logic and information, University of Oslo.

Volk, M. (2001). Exploiting the www as a corpus to resolve pp attachment ambiguities. In *Proceedings of Corpus Linguistics 2001*, Lancaster, UK.

Volk, M. (2002). Combining unsupervised and supervised methods for pp attachment disambiguation. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING-2002)*, Volume 2, Taipei, Taiwan, pp. 1065–1071.

Vossen, P. (Ed.) (1998). *EuroWordNet: A multilingual database with lexical semantic networks*. Kluwer Academic Publishers, Norwell.

Walker, M. A., A. K. Joshi, and E. F. Prince (1998). Centering in naturally occurring discourse: An overview. In A. K. J. Marilyn A. Walker and E. F. Prince (Eds.), *Centering Theory in Discourse*, pp. 1–30. Oxford University Press.

Webber, B. L. (1978). *A Formal Approach to Discourse Anaphora*. Ph. D. thesis, Harvard University.

Weiss, S. and C. Kulikowski (1991). *Computer systems that learn*. San Mateo, CA: Morgan Kaufmann.

White, A. P. and W. Z. Liu (1994). Bias in information-based measures in decision tree induction. *Machine Learning 15*(3), 321–329.

Whitelaw, C. and J. Patrick (2003). Named entity recognition using a character-based probabilistic approach. In *Proceedings of the Seventh Workshop on Computational Language Learning (CoNLL-2003)*, Edmonton, Canada.

Whittemore, G., K. Ferrara, and H. Brunner (1990). Empirical Study of Predictive Powers of Simple Attachment Schemes for Post-modifier Prepositional Phrases. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*.

Wilks, Y. (1975). A preferential, pattern-seeking semantics for natural language inference. *Artificial Intelligence 6*, 53–74.

Winograd, T. (1972). *Understanding Natural Language*. Orlando, FL, USA: Academic Press, Inc.

Wu, D., G. Ngai, and M. Carpuat (2003). A stacked, voted, stacked model for named entity recognition. In *Proceedings of the Seventh Workshop on Computational Language Learning (CoNLL-2003)*, Edmonton, Canada.

Wu, D., G. Ngai, M. Carpuat, J. Larsen, and Y. Yang (2002). Boosting for named entity recognition. In *Proceedings of the Sixth Workshop on Computational Language Learning (CoNLL-2002)*, Taipei, Taiwan.

Yang, X. and J. Su (2007). Coreference resolution using semantic relatedness information from automatically discovered patterns. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, Prague, pp. 528–535.

Yang, X., J. Su, and C. L. Tan (2005). Improving pronoun resolution using statistics-based semantic compatibility information. In *Proceedings of the 43rd Annual Meeting of the ACL*, Ann Arbor, pp. 165–172. Association for Computational Linguistics.

Yang, X., J. Su, and C. L. Tan (2006). Kernel-based pronoun resolution with structured syntactic knowledge. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*, Sydney, pp. 41–48. Association for Computational Linguistics.

Yang, X., S. Su, G. Zhou, and C. Tan (2004a). Improving pronoun resolution by incorporating coreferential information of candidates. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, Barcelona, Spain, pp. 128–135.

Yang, X., S. Su, G. Zhou, and C. Tan (2004b). A NP-cluster approach to coreference resolution. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-2004)*, Geneva, Switzerland.

Yang, X., G. Zhou, S. Su, and C. Tan (2003). Coreference resolution using competition learning approach. In *Proceedings of the 41th Annual Meeting of the Association for Compuatiational Linguistics (ACL-03),*, Sapporo, Japan, pp. 176–183.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, Cambridge, MA, pp. 189–196. ACM Press.

Zavrel, J. and W. Daelemans (1997). Memory-based learning: Using similarity for smoothing. In *Proceedings of the 35th annual meeting of the ACL*, Madrid.

Zavrel, J. and W. Daelemans (1999). Recent advances in memory-based part-of-speech tagging. In *VI Simposio Internacional de Comunicacion Social*, Santiago de Cuba, pp. 590–597.

Zavrel, J. and W. Daelemans (2003). *Text Mining, Theoretical Aspects and Applications*, Chapter Feature-rich memory-based classification for shallow NLP and information extraction, pp. 33–54. Springer Physica-Verlag.

Zavrel, J., W. Daelemans, and J. Veenstra (1997). Resolving PP Attachment Ambiguities with Memory-Based Learning. In T. Ellison (Ed.), *CoNLL97: Computational Natural Language Learning*, pp. 136–144. ACL.

Zhang, T. and D. Johnson (2003). A robust risk minimization based named entity recognition system. In *Proceedings of the Seventh Workshop on Computational Language Learning (CoNLL-2003)*, Edmonton, Canada.

Zhao, S. and D. Lin (2004). A nearest-neighbor method for resolving pp-attachment ambiguity. In *Proceedings of IJCNLP-04*.

Zhou, G. and J. Su (2002). Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, pp. 473–480.

Øvrelid, L. (2003). Subject or object? Syntactic disambiguation in Norwegian, stochastic Optimality Theory and application in an automatic system. Cand. philol. thesis in language, logic and information, University of Oslo.

Appendix A

Animate nouns found by mining the web

The tables in this appendix list the highest ranked animate nouns that were found by the offline web mining procedure described in chapter 6, using filtered results and only low-risk patterns.

The **A** column shows the number of times the noun was found in a pattern with *han* or *hun* as SUBJECT (i.e., in a context where an animate noun is expected), while the **I** column contains the corresponding frequency for patterns with *den* as SUBJECT (i.e., a context where we typically find inanimate nouns). The nouns are ranked according to the difference between the **A** and **I** columns. Only nouns for which **A**-**I** > 1 are included, meaning that only 1018 of the 4087 nouns that were found are listed. The rightmost column shows whether the noun was classified as correct ($\surd$) or incorrect ($\times$) in the evaluation described in section 6.6 (as explained in that section, only nouns that can *only* denote human beings were classified as correct).

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
1	rådgiver	264	1	√	39	koordinator	28	1	√
2	konsulent	196	1	√	40	advokatfullmektig	27	0	√
3	journalist	184	0	√	41	ingeniør	24	0	√
4	leder	184	1	√	42	reporter	24	0	√
5	prosjektleder	96	0	√	43	manager	24	1	√
6	assistent	83	0	√	44	avdelingsleder	23	0	√
7	førsteamanuensis	81	0	√	45	organist	22	0	√
8	professor	79	0	√	46	miljøarbeider	22	0	√
9	direktør	78	1	√	47	hushjelp	22	0	√
10	lege	68	1	√	48	vaktmester	21	0	√
11	advokat	63	0	√	49	lærling	21	0	√
12	sekretær	63	1	√	50	bedriftsrådgiver	21	0	√
13	sykepleier	59	0	√	51	regissør	21	0	√
14	stipendiat	57	0	√	52	representant	21	0	√
15	fotograf	51	0	√	53	tekstforfatter	20	0	√
16	lektor	50	0	√	54	amanuensis	20	0	√
17	instruktør	48	0	√	55	år	22	2	×
18	skuespiller	47	0	√	56	fysioterapeut	19	0	√
19	vikar	42	0	√	57	formann	19	0	√
20	kokk	41	0	√	58	illustratør	18	0	√
21	overlege	41	0	√	59	kunstner	18	0	√
22	musiker	39	0	√	60	førstekonsulent	18	0	√
23	høgskolelektor	39	0	√	61	misjonær	17	0	√
24	redaktør	39	0	√	62	salgssjef	17	0	√
25	rektor	39	0	√	63	programleder	17	0	√
26	prest	38	0	√	64	medlem	17	0	√
27	saksbehandler	35	0	√	65	dj	16	0	√
28	president	35	0	√	66	seniorrådgiver	16	0	√
29	servitør	34	0	√	67	seniorforsker	16	0	√
30	snekker	32	0	√	68	terapeut	16	0	√
31	tolk	32	0	√	69	assistentlege	16	0	√
32	produsent	31	0	√	70	bindeledd	22	7	√
33	mann	31	0	√	71	bibliotekar	15	0	√
34	frilansjournalist	30	0	√	72	modell	24	9	×
35	lærer	30	0	√	73	sosionom	15	0	√
36	i	31	2	×	74	billedkunstner	15	0	√
37	sjef	29	0	√	75	ordfører	16	1	√
38	psykolog	29	0	√	76	sang	14	0	×

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
77	økonomisjef	14	0	√	115	styreleder	10	0	√
78	seniorkonsulent	14	0	√	116	konge	10	0	√
79	nestleder	14	0	√	117	far	10	0	√
80	musikklærer	14	0	√	118	elektriker	9	0	√
81	universitetslektor	14	0	√	119	ekspeditør	9	0	√
82	offer	14	0	×	120	artist	9	0	√
83	spesialrådgiver	13	0	√	121	analytiker	9	0	√
84	sjåfør	13	0	√	122	freelancer	9	0	√
85	informasjonskonsulent	13	0	√	123	freelance	9	0	×
86	inspektør	13	0	√	124	frilans	9	0	×
87	helt	13	0	√	125	tannlege	9	0	√
88	hjelpepleier	13	0	√	126	konservator	9	0	√
89	komponist	13	0	√	127	dom	9	0	×
90	foredragsholder	13	0	√	128	førskolelærer	9	0	√
91	frisør	13	0	√	129	prosjektingeniør	9	0	√
92	helsesøster	13	0	√	130	jordmor	9	0	√
93	dirigent	13	0	√	131	jurist	9	0	√
94	kommunikasjonsrådgiver	12	0	√	132	dans	9	0	×
95	freelancemusiker	12	0	√	133	kontorsjef	9	0	√
96	mekaniker	12	0	√	134	kritiker	9	0	√
97	guide	16	4	√	135	informasjonsrådgiver	9	0	√
98	agent	13	1	√	136	regiassistent	9	0	√
99	kaptein	12	0	√	137	markedsdirektør	9	0	√
100	informasjonssjef	11	0	√	138	konserndirektør	9	0	√
101	prosjektkoordinator	11	0	√	139	arkitekt	9	0	√
102	aksjemegler	11	0	√	140	avdelingsdirektør	9	0	√
103	dommerfullmektig	11	0	√	141	avdelingsingeniør	9	0	√
104	mellommann	11	0	√	142	konsernsjef	9	0	√
105	sønn	11	0	√	143	person	9	0	√
106	avdelingssjef	10	0	√	144	it-konsulent	8	0	√
107	allmennlege	10	0	√	145	nattevakt	8	0	√
108	dagmamma	10	0	√	146	anestesisykepleier	8	0	√
109	distriktsmusiker	10	0	√	147	kirurg	8	0	√
110	språklærer	10	0	√	148	regnskapsfører	8	0	√
111	post	10	0	×	149	markedssjef	8	0	√
112	ekspert	10	0	√	150	informasjonsleder	8	0	√
113	kommunelege	10	0	√	151	kateket	8	0	√
114	generalsekretær	10	0	√	152	forskningsassistent	8	0	√

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
153	bartender	8	0	√	191	arrangør	7	0	√
154	vaskehjelp	8	0	√	192	logistikksjef	7	0	√
155	overingeniør	8	0	√	193	gårdsgutt	7	0	√
156	faglærer	8	0	√	194	budeie	7	0	√
157	politi	9	1	√	195	forsker	7	0	√
158	morsmålslærer	8	0	√	196	pedagog	7	0	√
159	renholder	8	0	√	197	konsertmester	7	0	√
160	produktsjef	8	0	√	198	livvakt	7	0	√
161	personalkonsulent	8	0	√	199	kurator	7	0	√
162	høyskolelektor	8	0	√	200	partner	7	0	√
163	spesialist	8	0	√	201	kvinne	7	0	√
164	pianist	8	0	√	202	gift	7	0	√
165	solist	8	0	√	203	datter	7	0	√
166	sjømann	8	0	√	204	død	8	1	×
167	rådmann	8	0	√	205	bilmekaniker	6	0	√
168	spiller	9	1	√	206	dørvakt	6	0	√
169	kontaktperson	8	0	√	207	revisor	6	0	√
170	politidirektør	8	0	√	208	stylist	6	0	√
171	menneske	8	0	√	209	maskinfører	6	0	√
172	venn	8	0	√	210	pastor	6	0	√
173	kjøkkensjef	7	0	√	211	medarbeider	6	0	√
174	frilansmusiker	7	0	√	212	bygningsarbeid	6	0	×
175	tekniker	7	0	√	213	sangpedagog	6	0	√
176	fotojournalist	7	0	√	214	økonom	6	0	√
177	politimann	7	0	√	215	anleggsarbeider	6	0	√
178	underdirektør	7	0	√	216	predikant	6	0	√
179	butikksjef	7	0	√	217	ambassadør	6	0	√
180	spion	7	0	√	218	vit.ass.	6	0	√
181	fastlege	7	0	√	219	husholderske	6	0	√
182	sekretariatsleder	7	0	√	220	salgskonsulent	6	0	√
183	kulturjournalist	7	0	√	221	kantor	6	0	√
184	skribent	7	0	√	222	frilansfotograf	6	0	√
185	nyhetsreporter	7	0	√	223	danselærer	6	0	√
186	utviklingslede	7	0	√	224	seksjonsleder	6	0	√
187	kontorleder	7	0	√	225	prosjekt	6	0	√
188	kontorist	7	0	√	226	smed	6	0	√
189	manusforfatter	7	0	√	227	resepsjonist	6	0	√
190	engelsklærer	7	0	√	228	reklamefotograf	6	0	√

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
229	industriarbeider	6	0	√	267	litteraturkritiker	5	0	√
230	frilansskribent	6	0	√	268	tjenestepike	5	0	√
231	forlagsredaktør	6	0	√	269	kontorfullmektig	5	0	√
232	scenograf	6	0	√	270	flyvertinne	5	0	√
233	skrift-designer	6	0	√	271	psykiater	5	0	√
234	finansanalytiker	6	0	√	272	typograf	5	0	√
235	programutvikler	6	0	√	273	kontordame	5	0	√
236	navigatør	7	1	√	274	kokke	5	0	√
237	allmennpraktiker	6	0	√	275	barnelege	5	0	√
238	forskningsleder	6	0	√	276	barnehageassistent	5	0	√
239	gruppeleder	6	0	√	277	kundekonsulent	5	0	√
240	filmanmelder	6	0	√	278	stuepike	5	0	√
241	programsekretær	6	0	√	279	koreograf	5	0	√
242	næringslivsjournalist	6	0	√	280	integreringskonsulent	5	0	√
243	senior	6	0	√	281	anleggsleder	5	0	√
244	historiker	6	0	√	282	sjefskonsulent	5	0	√
245	kapellan	6	0	√	283	produksjonsleder	5	0	√
246	forretningsfører	6	0	√	284	butikkmedarbeider	5	0	√
247	sceneinstruktør	6	0	√	285	politiker	5	0	√
248	forfatter	6	0	√	286	systemkonsulent	5	0	√
249	spesialkonsulent	6	0	√	287	personalsjef	5	0	√
250	naturfotograf	6	0	√	288	helserådgiver	5	0	√
251	produksjonssjef	6	0	√	289	motefotograf	5	0	√
252	sjelesørger	6	0	√	290	astrolog	5	0	√
253	lensmann	6	0	√	291	maskinist	5	0	√
254	statsminister	6	0	√	292	skomaker	5	0	√
255	dronning	6	0	×	293	norsklærer	5	0	√
256	markedskonsulent	5	0	√	294	nyhetsjournalist	5	0	√
257	drosjesjåfør	5	0	√	295	doktorgradsstipendiat	5	0	√
258	it	5	0	×	296	studieveileder	5	0	√
259	sikkerhetsvakt	5	0	√	297	sosialarbeider	5	0	√
260	systemutvikler	5	0	√	298	bedriftslege	5	0	√
261	kursleder	5	0	√	299	timelærer	5	0	√
262	montør	5	0	√	300	distriktssekretær	5	0	√
263	førstelektor	5	0	√	301	jordskiftedommer	5	0	√
264	styremedlem	5	0	√	302	enhetslede	5	0	√
265	advokatassistent	5	0	√	303	adjunkt	5	0	√
266	kassadame	5	0	√	304	keepertrener	5	0	√

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
305	studierådgiver	5	0	√	343	prosjektmedarbeider	4	0	√
306	sjefingeniør	5	0	√	344	produksjonsassistent	4	0	√
307	ekspedisjonssjef	5	0	√	345	fagsjef	4	0	√
308	landsdelsmusiker	5	0	√	346	miljøterapeut	4	0	√
309	døråpner	8	3	×	347	fylkessekretær	4	0	√
310	regent	6	1	√	348	senioringeniør	4	0	√
311	mor	5	0	√	349	innkjøpssjef	4	0	√
312	fylkeslege	5	0	√	350	finansdirektør	4	0	√
313	operasjonsleder	5	0	√	351	ambulansesjåfør	4	0	√
314	vitne	7	2	√	352	director	4	0	√
315	software	4	0	×	353	tømmerhogger	4	0	√
316	kundebehandler	4	0	√	354	bygningssnekker	4	0	√
317	seminarleder	4	0	√	355	seksjonsoverlege	4	0	√
318	vakt	4	0	√	356	evangelist	4	0	√
319	lærervikar	4	0	√	357	rørlegger	4	0	√
320	medierådgiver	4	0	√	358	hybridkunstner	4	0	√
321	spillerutvikler	4	0	√	359	gartner	4	0	√
322	håndverker	4	0	√	360	programkoordinator	4	0	√
323	styrmann	4	0	√	361	gud	4	0	√
324	bonde	4	0	√	362	postmester	4	0	√
325	vernepleie	4	0	×	363	forskningssjef	4	0	√
326	gjesteforeleser	4	0	√	364	diakon	4	0	√
327	teamleder	4	0	√	365	huslærer	4	0	√
328	turistguide	4	0	√	366	teaterfotograf	4	0	√
329	massør	4	0	√	367	frilansdanser	4	0	√
330	kultursjef	4	0	√	368	folkehøgskolelærer	4	0	√
331	fengselsbetjent	4	0	√	369	menighetssekretær	4	0	√
332	investeringsrådgiver	4	0	√	370	økonomirådgiver	4	0	√
333	sjefredaktør	4	0	√	371	kontorassistent	4	0	√
334	tømmerhugger	4	0	√	372	syerske	4	0	√
335	formgiver	4	0	√	373	tysklærer	4	0	√
336	sosialkonsulent	4	0	√	374	telemontør	4	0	√
337	frilanssanger	4	0	√	375	distriktsarbeidssjef	4	0	√
338	ernæringsfysiolog	4	0	√	376	offiser	4	0	√
339	konduktør	4	0	√	377	byrådssekretær	4	0	√
340	hjemmehjelp	4	0	√	378	men	4	0	×
341	bioingeniør	4	0	√	379	dosent	4	0	√
342	bussjåfør	4	0	√	380	styreformann	4	0	√

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
381	kunstfotograf	4	0	√	419	fagarbeid	3	0	×
382	operatør	4	0	√	420	forretningsrådgiver	3	0	√
383	divisjonsdirektør	4	0	√	421	regel	9	6	×
384	utsending	4	0	√	422	konferansier	3	0	√
385	universitetsstipendiat	4	0	√	423	musikkterapeut	3	0	√
386	kapellmester	4	0	√	424	forretningsutvikler	3	0	√
387	sokneprest	4	0	√	425	avdelingsoverlege	3	0	√
388	restaurantsjef	4	0	√	426	bokanmelder	3	0	√
389	gjesteprofessor	4	0	√	427	inspirator	4	1	√
390	økonomidirektør	4	0	√	428	kameramann	3	0	√
391	guvernante	4	0	√	429	programvareutvikler	3	0	√
392	politijurist	4	0	√	430	ungdomsarbeider	3	0	√
393	skoleinspektør	4	0	√	431	gitarlærer	3	0	√
394	stortingsrepresentant	4	0	√	432	attføringslede	3	0	√
395	vararepresentant	4	0	√	433	teaterpedagog	3	0	√
396	varaordfører	4	0	√	434	hest	3	0	×
397	kapitalist	4	0	√	435	art	3	0	×
398	samtalepartner	4	0	√	436	regionsjef	3	0	√
399	talsmann	4	0	√	437	prestevikar	3	0	√
400	ressursperson	4	0	√	438	freelancejournalist	3	0	√
401	frontfigur	4	0	√	439	pilot	3	0	√
402	blanding	5	1	×	440	miljøvernleder	3	0	√
403	sogneprest	4	0	√	441	webdesigner	3	0	√
404	soldat	4	0	√	442	programskaper	3	0	√
405	pikespiller	4	0	√	443	praktikant	3	0	√
406	naturtalent	4	0	√	444	lærerinne	3	0	√
407	børsmegler	4	0	√	445	lobbyist	3	0	√
408	lastebilsjåfør	3	0	√	446	fotomodell	3	0	√
409	kontrollør	3	0	√	447	sivilarkitekt	3	0	√
410	volontør	3	0	√	448	farmasøyt	3	0	√
411	gullsmed	3	0	√	449	trener	3	0	√
412	fagkonsulent	3	0	√	450	skattejurist	3	0	√
413	kelner	3	0	√	451	gudinne	3	0	√
414	truckfører	3	0	√	452	allmennlærer	3	0	√
415	prosjektsekretær	3	0	√	453	gravferdskonsulent	3	0	√
416	renovatør	3	0	√	454	grafiker	3	0	√
417	barnehageonkel	3	0	√	455	sykesøster	3	0	√
418	ryddegutt	3	0	√	456	spesialpsykolog	3	0	√

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
457	høgskolelærer	3	0	√	495	programsjef	3	0	√
458	støttekontakt	3	0	√	496	business	3	0	√
459	klesselger	3	0	√	497	korist	3	0	√
460	musikkpedagog	3	0	√	498	studiomusiker	3	0	√
461	plansjef	3	0	√	499	serviceingeniør	3	0	√
462	undervisningskonsulent	3	0	√	500	salgsdirektør	3	0	√
463	reklametegner	3	0	√	501	golfinstruktør	3	0	√
464	møbeldesigner	3	0	√	502	sosialleder	3	0	√
465	skiftleder	3	0	√	503	delegat	3	0	√
466	ekstrahjelp	3	0	√	504	utdanningskonsulent	3	0	√
467	kirkeverge	3	0	√	505	organisasjonssekretær	3	0	√
468	salgsjef	3	0	√	506	dansepedagog	3	0	√
469	tilkallingsvikar	3	0	√	507	seksjonssjef	3	0	√
470	prosjektdirektør	3	0	√	508	observatør	3	0	√
471	blikkenslager	3	0	√	509	revisormedarbeider	3	0	√
472	almenpraktiker	3	0	√	510	politiadvokat	3	0	√
473	service	4	1	×	511	rådgiver/prosjektleder	3	0	√
474	sjefsprodusent	3	0	√	512	webredaktør	3	0	√
475	skredder	3	0	√	513	hjelpearbeider	3	0	√
476	filmskaper	3	0	√	514	dramaturg	3	0	√
477	ad-assistent	3	0	√	515	frilansmusikar	3	0	√
478	reklamefilmregissør	3	0	√	516	akkompagnatør	3	0	√
479	tømmermann	3	0	√	517	avd.	3	0	×
480	korrespondent	3	0	√	518	lærebokforfatter	3	0	√
481	kunstkritiker	3	0	√	519	frilansskuespiller	3	0	√
482	informasjonsarkitekt	3	0	√	520	oppsynsmann	3	0	√
483	vertinne	3	0	√	521	markedsrådgiver	3	0	√
484	fabrikkarbeider	3	0	√	522	storyteller	3	0	√
485	fløytelærer	3	0	√	523	sauegjeter	3	0	√
486	spåkone	3	0	√	524	verksmester	3	0	√
487	advokatsekretær	3	0	√	525	postdoktor	3	0	√
488	it-journalist	3	0	√	526	forhørsdom	3	0	√
489	logoped	3	0	√	527	tjenestejente	3	0	√
490	klassestyrer	3	0	√	528	postdoktorstipendiat	3	0	√
491	studiekonsulent	3	0	√	529	universitetslærer	3	0	√
492	museumspedagog	3	0	√	530	gårdbruker	3	0	√
493	økonomikonsulent	3	0	√	531	moderator	3	0	√
494	postbud	3	0	√	532	andre	3	0	×

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
533	coole	3	0	×	571	økolog	2	0	√
534	studiesjef	3	0	√	572	liaisonoffiser	2	0	√
535	undervisningsinspektør	3	0	√	573	eures	2	0	×
536	dekan	3	0	√	574	fjernsynsfotograf	2	0	√
537	klarinettist	3	0	√	575	skulptør	2	0	√
538	ungdomssekretær	3	0	√	576	teaterinstruktør	2	0	√
539	teaterdikter	3	0	√	577	ambulansepersonell	2	0	√
540	prorektor	3	0	√	578	angiver	2	0	√
541	varamedlem	3	0	√	579	filmregissør	2	0	√
542	pave	3	0	√	580	promo-produsent	2	0	√
543	diktator	3	0	√	581	asyladvokat	2	0	√
544	forteller	3	0	√	582	obligasjonsmegler	2	0	√
545	talsperson	3	0	√	583	slave	3	1	√
546	elev	3	0	√	584	bokbinder	2	0	√
547	kone	3	0	√	585	eiendomsmekler	2	0	√
548	hermia	3	0	√	586	personelloffiser	2	0	√
549	mentor	3	0	√	587	architect	2	0	√
550	læremester	3	0	√	588	kroppsøvingslærer	2	0	√
551	forbilde	3	0	√	589	fotballtrener	2	0	√
552	bestefar	3	0	√	590	datakonsulent	2	0	√
553	medium	3	0	×	591	salgsfremmer	2	0	√
554	stedfortreder	3	0	√	592	it-arkitekt	2	0	√
555	vokalist	3	0	√	593	gynekolog	2	0	√
556	biskop	3	0	√	594	utviklingssjef	2	0	√
557	eier	3	0	√	595	hovedtrener	2	0	√
558	pappa	3	0	√	596	kranfører	2	0	√
559	businessmann	3	0	√	597	vekt	2	0	×
560	soloartist	3	0	√	598	dyr	2	0	×
561	styrer	3	0	√	599	idrettslege	2	0	√
562	skikkelse	3	0	√	600	sivilarbeider	2	0	√
563	dame	3	0	√	601	personalrådgiver	2	0	√
564	slektning	3	0	√	602	turnemanager	2	0	√
565	etterkommer	3	0	×	603	kontormedarbeider	2	0	√
566	deltager	3	0	√	604	nettlærer	2	0	√
567	storhusholdningsgrossist	2	0	√	605	sjakklærer	2	0	√
568	møbelsnekker	2	0	√	606	produksjonstekniker	2	0	√
569	pianostemmer	2	0	√	607	brannmann	2	0	√
570	funksjonær	2	0	√	608	ekspertkommentator	2	0	√

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
609	miljørådgiver	2	0	√	647	fagmedarbeider	2	0	√
610	reklametekniker	2	0	√	648	barneskolelærer	2	0	√
611	skogsarbeider	2	0	√	649	hjelpelærer	2	0	√
612	hmsk-konsulent	2	0	√	650	finanskonsulent	2	0	√
613	valgkamp-medarbeider	2	0	√	651	doktorgradstipendiat	2	0	√
614	driftsingeniør	2	0	√	652	informasjonsmedarbeider	2	0	√
615	frilancemusik	2	0	√	653	driftskonsulent	2	0	√
616	miljøforskningskonsulent	2	0	√	654	sportsjournalist	2	0	√
617	driftssjef	2	0	√	655	storyboardtegner	2	0	√
618	politibetjent	2	0	√	656	produktutvikler	2	0	√
619	eiendomsmegler	2	0	√	657	apotektekniker	2	0	√
620	sykkelbud	2	0	√	658	sosialkurator	2	0	√
621	idrettsfredskorps	2	0	√	659	fyrbøter	2	0	√
622	barnevernskonsulent	2	0	√	660	matros	2	0	√
623	bedriftskonsulent	2	0	√	661	controlle	2	0	√
624	healer	2	0	√	662	disponent	2	0	√
625	automatik	2	0	√	663	postmann	2	0	√
626	skiinstruktør	2	0	√	664	røykdykker	2	0	√
627	dekoratør	2	0	√	665	jr.	2	0	√
628	aupair	2	0	√	666	jazzmusiker	2	0	√
629	nettverkspesialist	2	0	√	667	visergutt	2	0	√
630	massasjeterapeut	2	0	√	668	roadie	2	0	√
631	ettåring	2	0	×	669	musikkjournalist	2	0	√
632	barnevernspedagog	2	0	√	670	finansrådgiver	2	0	√
633	veterinær	2	0	√	671	distriktssjef	2	0	√
634	dramalærer	2	0	√	672	lydtekniker	2	0	√
635	dramapedagog	2	0	√	673	høvleriarbeider	2	0	√
636	servitrise	2	0	√	674	utdanningsrådgiver	2	0	√
637	negledesigner	2	0	√	675	deltidskirurg	2	0	√
638	husøkonom	2	0	√	676	web-redaktør	2	0	√
639	fløytist	2	0	√	677	junior	2	0	√
640	barnepleier	2	0	√	678	skogbrukslede	2	0	√
641	omsorgsarbeid	2	0	×	679	byssegutt	2	0	√
642	badevakt	2	0	√	680	jr	2	0	√
643	instituttleder	2	0	√	681	spesiallærer	2	0	√
644	glamourmodell	2	0	√	682	legevikar	2	0	√
645	oversetter	3	1	√	683	management	2	0	√
646	reservoaringeniør	2	0	√	684	visesanger/skuespiller	2	0	√

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
685	bookingagent	2	0	√	723	svømmeinstruktør	2	0	√
686	ingenør	2	0	√	724	kommunefysioterapeut	2	0	√
687	politiavdelingsjef	2	0	√	725	studentkoordinator	2	0	√
688	stillasarbeider	2	0	√	726	interaksjonsdesigner	2	0	√
689	lydmann	2	0	√	727	markedskoordinator	2	0	√
690	analyst	2	0	√	728	analytiker/controlle	2	0	√
691	walstad	2	0	√	729	skoleleder	2	0	√
692	hjulmaker	2	0	√	730	koreografiassistent	2	0	√
693	aktuarkonsulent	2	0	√	731	medisinkvinne	2	0	√
694	fagleder	2	0	√	732	freelansemusiker	2	0	√
695	sykkelfabrikant	2	0	√	733	legesekretær	2	0	√
696	grafisk	2	0	×	734	krimreporter	2	0	√
697	korpsdirigent	2	0	√	735	nyhetsanker	2	0	√
698	læregutt	2	0	√	736	vp.	2	0	√
699	barmanager	2	0	√	737	sosiolog	2	0	√
700	kursholder	2	0	√	738	trell	2	0	√
701	visumattaché	2	0	√	739	kongressjef	2	0	√
702	sushi-kokk	2	0	√	740	tv-produsent	2	0	√
703	arbeidskonsulent	2	0	√	741	smådyrpraktiker	2	0	√
704	test	2	0	×	742	ergoterapeut	2	0	√
705	flygerinspektør	2	0	√	743	associate	2	0	×
706	it-.	2	0	×	744	pasientvenn	2	0	√
707	investeringsdirektør	2	0	√	745	produksjonssekretær	2	0	√
708	klimaforsker	2	0	√	746	komiker	2	0	√
709	markedskreatør	2	0	√	747	barnevakt	2	0	√
710	delegato	2	0	×	748	mellomleder	2	0	√
711	marketingssjef	2	0	√	749	au-pair	2	0	√
712	motor	3	1	×	750	prosjektkonsulent	2	0	√
713	næringsrådgiver	2	0	√	751	prosjektassistent	2	0	√
714	kordirigent	2	0	√	752	forretningsadvokat	2	0	√
715	distriktsrepresentant	2	0	√	753	programmedarbeider	2	0	√
716	museumskonsulent	2	0	√	754	familieterapeut	2	0	√
717	bergskriver	2	0	√	755	forskningskonsulent	2	0	√
718	karosserimaker/restaurer	2	0	√	756	sjefstrateg	2	0	√
719	krigskorrespondent	2	0	√	757	consultant	2	0	√
720	sakfører	2	0	√	758	professorstipendiat	2	0	√
721	utlendingsattaché	2	0	√	759	innspillingsleder	2	0	√
722	oppdragsforsker	2	0	√	760	konsulent/prosjektleder	2	0	√

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
761	skipper	2	0	√	799	sykehusprest	2	0	√
762	forlagskonsulent	2	0	√	800	ekspeditrise	2	0	√
763	gitarist	2	0	√	801	senterkoordinator	2	0	√
764	toppsjef	2	0	√	802	tid	2	0	×
765	statist	2	0	√	803	kommunikasjonsforvalter	2	0	√
766	producer	2	0	√	804	4h-konsulent	2	0	√
767	elektromontør	2	0	√	805	byråkrat	2	0	√
768	utekontakt	2	0	√	806	kjøkkenassistent	2	0	√
769	teatersjef	2	0	√	807	sexolog	2	0	√
770	mediekonsulent	2	0	√	808	pressefotograf	2	0	√
771	aktivitør	2	0	√	809	yogalærer	2	0	√
772	treningsveileder	2	0	√	810	leselærer	2	0	√
773	livredder	2	0	√	811	finansmedarbeider	2	0	√
774	spesialpedagog	2	0	√	812	aksje	2	0	×
775	kompetanserådgiver	2	0	√	813	sjefsingeniør	2	0	√
776	kunstformidler	2	0	√	814	fylkesarkeolog	2	0	√
777	industridesign	2	0	√	815	fritidsklubbmedarbeider	2	0	√
778	sommervikar	2	0	√	816	aksjemegler/meglersjef	2	0	√
779	portrettfotograf	2	0	√	817	golfbanearkitekt	2	0	√
780	selger	2	0	√	818	fransklærer	2	0	√
781	gjesteforsker	2	0	√	819	fotballagent	2	0	√
782	forskningsdirektør	2	0	√	820	sjefsøkonom	2	0	√
783	elektrikar	2	0	√	821	markedsføringsrådgiver	2	0	√
784	billettkontrollør	2	0	√	822	inspeksjonslede	2	0	√
785	bilfotograf	2	0	√	823	integreringsmedarbeid	2	0	√
786	motivasjonstrener	2	0	√	824	driftstekniker	2	0	√
787	byggeleder	2	0	√	825	fiolinist	2	0	√
788	astronom	2	0	√	826	slagverker	2	0	√
789	kirkemusiker	2	0	√	827	solo	2	0	×
790	prosessingeniør	2	0	√	828	matkonsulent	2	0	√
791	arkivar	2	0	√	829	stadskonduktør	2	0	√
792	sikkerhetskonsulent	2	0	√	830	kommentator	2	0	√
793	kunstpedagog	2	0	√	831	spesialagent	2	0	√
794	førstekonservator	2	0	√	832	kontroll-lege	2	0	√
795	forskar	2	0	√	833	kulturformidler	2	0	√
796	forskningsstipendiat	2	0	√	834	tannkirurg	2	0	√
797	psykologspesialist	2	0	√	835	bokillustratør	2	0	√
798	akupunktør	2	0	√	836	reparatør	2	0	√

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
837	administrasjons-sjef	2	0	√	875	hjerneforsker	2	0	√
838	handelsmann	2	0	√	876	rengjører	2	0	√
839	takstmann	2	0	√	877	klasseforstander	2	0	√
840	omsorgsassistent	2	0	√	878	sykehuslege	2	0	√
841	fjelloppsyn	2	0	×	879	varemerkekonsulent	2	0	√
842	omviser	2	0	√	880	formingslærer	2	0	√
843	kulturhussjef	2	0	√	881	magister	2	0	√
844	omgangsskolelærer	2	0	√	882	familiekonsulent	2	0	√
845	utenriksmedarbeider	2	0	√	883	forskningskoordinator	2	0	√
846	etablererveileder	2	0	√	884	personaldirektør	2	0	√
847	poet	2	0	√	885	ungdomspastor	2	0	√
848	arkeolog	2	0	√	886	kampanjesekretær	2	0	√
849	produksjonskeramiker	2	0	√	887	markedsanalytiker	2	0	√
850	lærer/lektor	2	0	√	888	politifullmektig	2	0	√
851	fiskeoppsyn	2	0	×	889	seniorøkonom	2	0	√
852	mekaniker/maskinarbeider	2	0	√	890	strategirådgiver	2	0	√
853	utmarkskonsulent	2	0	√	891	vikarlege	2	0	√
854	uteleder	2	0	√	892	operasjonssykepleier	2	0	√
855	it-sjef	2	0	√	893	tannpleie	2	0	×
856	dekorasjonsmale	2	0	√	894	kommunejordmor	2	0	√
857	dreng	2	0	√	895	sikkerhetsrådgiver	2	0	√
858	nestkommanderende	2	0	√	896	sjefsinstruktør	2	0	√
859	edb	2	0	×	897	leder/musikkonsulent	2	0	√
860	sersjant	2	0	√	898	fagottist	2	0	√
861	kulturredaktør	2	0	√	899	regionleder	2	0	√
862	ordenspoliti	2	0	√	900	brannsjef	2	0	√
863	workshop	2	0	×	901	fou-direktør	2	0	√
864	frilansscenograf	2	0	√	902	domorganist	2	0	√
865	tannhelsesekretær	2	0	√	903	hjelpetrener	2	0	√
866	redaksjonsekretær	2	0	√	904	formidlingslede	2	0	√
867	sentralbordmedarbeider	2	0	√	905	technology	2	0	×
868	trombone-pedagog	2	0	√	906	herredsagronom	2	0	√
869	tegner	2	0	√	907	konstruksjonstegner	2	0	√
870	pleie	2	0	×	908	solo-bratsjist	2	0	√
871	seksjonssjef/rådgiver	2	0	√	909	økonomimedarbeider	2	0	√
872	ungdomsprest	2	0	√	910	fullmektig	2	0	√
873	belysningsrådgiver	2	0	√	911	admiral	2	0	√
874	lydingeniør	2	0	√	912	huskoreograf	2	0	√

Rank	Lemma	A	I	OK?	Rank	Lemma	A	I	OK?
913	1.konsertmester	2	0	√	951	surrogatmor	2	0	√
914	steward	2	0	√	952	venus	2	0	√
915	trompetist	2	0	√	953	mamma	2	0	√
916	styrerepresentant	2	0	√	954	skolesjef	2	0	√
917	føstekeeper	2	0	√	955	satiriker	2	0	√
918	verv	2	0	×	956	autoritet	2	0	√
919	klubbleder	2	0	√	957	fadder	2	0	√
920	portvakt	2	0	√	958	vokter	2	0	√
921	visepresident	2	0	√	959	biperson	2	0	√
922	adm.dir.	2	0	√	960	overdommer	2	0	√
923	overlærer	2	0	√	961	figur	2	0	×
924	vice	2	0	√	962	giftekniv	2	0	√
925	midtpunkt	2	0	×	963	quisling	2	0	√
926	marvin	2	0	√	964	guvernør	2	0	√
927	kong	2	0	√	965	humorist	2	0	√
928	hustru	2	0	√	966	kar	2	0	√
929	anna	2	0	√	967	glittrende	2	0	×
930	unge	2	0	×	968	stallgutt	2	0	√
931	omsorgsperson	2	0	√	969	drukkenbolt	2	0	√
932	føresett	2	0	√	970	vitsemaker	2	0	√
933	riksforstander	2	0	√	971	foregangsmann	2	0	√
934	allah	2	0	√	972	folkeopplysningsmann	2	0	√
935	klasselærer	2	0	√	973	amatørskuespiller	2	0	√
936	budbringer	2	0	√	974	fyr	2	0	√
937	turneringsleder	2	0	√	975	herre	2	0	√
938	ventilasjon	2	0	×	976	jeger	2	0	√
939	ansikt	2	0	×	977	dommer	2	0	√
940	fødselshjelper	2	0	√	978	cosmopolitan-tilhenger	2	0	√
941	motivator	2	0	√	979	enke	2	0	√
942	altmuligmann	2	0	√	980	pike	2	0	√
943	talkshowvert	2	0	√	981	kongsdatter	2	0	√
944	sendebud	2	0	√	982	scenekunstner	2	0	√
945	torpedo	2	0	√	983	bror	2	0	√
946	sheriff	2	0	√	984	legende	2	0	√
947	onkel	2	0	√	985	ektemann	2	0	√
948	sysselmann	2	0	√	986	kristen	2	0	√
949	informant	2	0	√	987	nr.	2	0	×
950	horemamma	2	0	√	988	arbeidshest	2	0	×

Rank	Lemma	A	I	OK?
989	høyreback	2	0	√
990	størrelse	2	0	×
991	prins	2	0	√
992	konkurransehund	2	0	×
993	statsministerkandidat	2	0	√
994	kvitt	2	0	×
995	rockeband	2	0	√
996	datanerd	2	0	√
997	student	2	0	√
998	ungpike	2	0	√
999	fraggel	2	0	×
1000	blitzer	2	0	√
1001	tadkom-kriger	2	0	√
1002	kort	2	0	×
1003	flott	2	0	×
1004	forskjell	2	0	×
1005	velsignelse	2	0	×
1006	måned	2	0	×
1007	beskjed	2	0	×
1008	fugl	2	0	×
1009	råd	2	0	√
1010	livstre	2	0	×
1011	inkarnasjon	2	0	√
1012	verande	2	0	×
1013	hvalforsker	2	0	√
1014	versting	2	0	√
1015	big-man	2	0	√
1016	bokser	2	0	√
1017	pokerspiller	2	0	√
1018	terrorist	2	0	√